CONFIGURATION DESIGN OPTIMIZATION METHOD

A Dissertation

Presented to

the Graduate School of

Clemson University

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

Mechanical Engineering

---

by

Pierre M. Grignon

December 1998

Advisor: Dr. Georges M. Fadel

ABSTRACT

During the product design process, the optimization of the parameters (dimensions, shapes, materials) has been, until recently, the exclusive domain of the part design stage. This optimization is now spreading into two directions: first, to the conceptual design stage during which approximated mathematical models are solved to find the global parameters driving the design; second to the sub-assembly and assembly design stage during which the components are united to form the final product. It is only at this stage that system level characteristics, strongly linked to the desire of the customers, can be checked. These characteristics, considered as objectives by the engineer, and reflecting different knowledge fields, link parameters at various levels of the system and thus break the well established hierarchical organization of the full system. At present, no satisfactory solution is implemented to optimize these additional relations that transform the assembly design from a constraint satisfaction problem into a non-linear constrained multi objective optimization problem.

Hence, this work presents a method for optimizing system level assembly characteristics of complex mechanical assemblies by placement of freeform components. This method finds multiple solutions to the Configuration Design Problem and proposes a simple cooperation scheme with the engineer.

The review of the industrial applications showed that no assumptions could be made on the type of components and systems. However, system optimization usually occurred after components shapes and their functional links were settled. From the mathematical representation point of view, the variables can then be restricted to displacements and can be considered as continuous in all cases.

Once this is assumed, three areas are necessary to address the goal:

- define the complexity of an Engineering Configuration Design Problem (ECDP),
- define the method
- measure its performance.

In order to address the first goal, a mathematical formulation is adopted, that, unlike many formulations found in the literature review, could address any type of components (involving non convex, hollow, sharp edges components). This formulation relies on the use of continuous variables for defining the authorized movements of the components with respect to each others and on penalty functions for penalizing unfeasible configurations. Then, based on the notion of function landscape, four criteria are chosen to answer the first need i.e. evaluating the complexity of the ECDPs. Trend, Roughness, Dilution of the feasible area and Dilution of the solution were defined for any type of C^1 functions and linked to simple characteristics of the ECDPs (size and number of the components, size of the system).

The second goal, the definition of the method, is addressed as finding multiple global extrema of a non-linear optimization problem. A Genetic Algorithm working on population of sets instead of population of individual points is proposed to search the variables space and to provide multiple solutions for three reasons. First, the GA works on several designs at the same time; second, it is able to deal with highly non-linear functions; and third it capitalizes on the knowledge acquired at previous trials. The analysis of the complexity classification of some well-known packing problems provided clues for proposing three enhancements to this initial method. First, the penalty should always be combined with the objective function for unfeasible configurations evaluation, then, the use of a Local Search, and Relative Placement contribute to decrease the CDP's complexity. Eventually, to increase the chances of escaping local minima, an adaptive range strategy was suggested.

Before verifying the validity of these choices, some criteria had to be designed to measure the quality of the Pareto set obtained using the CDOM. Three criteria are chosen to measure the quality of the set itself (distance from the extreme planes, distribution of the points, flatness) and two for measuring the performance of the CDOM (speed and repeatability).

A series of cubes CDPs and two engineering cases were submitted to the method for validating the enhancements and the complexity classification. The conclusion of this part of the study pointed out that the Local Search and Range Adaptation enhancements were the most effective. Relative placement did not bring a clear advantage over the base strategy when dealing with CDPs in which the relative motion is not linked to a functional need.

Among the two types of criteria proposed for first rating the CDPs complexity and second for rating the CDOM, only the former were clearly satisfactory and can be reused in further studies. To assess the quality of a Pareto set, several criteria should be rethought.

The CDOM successfully produces multiple solutions to the test cases submitted. Its drop in performance is linear while the CDPs complexity increases exponentially. It also proved to be able to handle any type of components. The lack of assumptions made on the type of objective functions makes it robust for handling continuous and discrete objectives. Working with displacements on one hand and, on the other hand, having the possibility to choose which components can move and which remain static, helps the engineer to keep a close contact with the optimization process. The limitations of the method concern the low number of moving components that can be taken into account at the same time (less than 10 system components using relative placement).

DEDICATION

I dedicate this work to my family.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Georges M. Fadel, for all his assistance and guidance. His high expectations and continuous support are greately appreciated. I am grateful to the other committee members: Dr. Wei Chen, Dr. Richard Figliola, Dr. Roy Pargas, and Dr. Malgorzata Wiecek.

I am grateful to Stephane Morvan and Jean-François Fayolle for their support while I was back in France and still writing this dissertation.

I would like to thank several members of the Clemson Research in Engineering Design and Optimization group who helped me to put this work together and especially Venugopal Challa.

Finally, I would like to thank Johnny Fadel, Marc Long and Anthony Sannino for their support when I was far from my family.

TABLE OF CONTENTS

Page

TITLE PAGE i

ABSTRACT ii

DEDICATION vi

ACKNOWLEDGMENTS vii

LIST OF TABLES x

LIST OF FIGURES xii

CHAPTER

Some Considerations on Design 1
The Steps of Design 3
Motivations 9
Goal of this Dissertation 14
The Fundamental Bases 14
Fundamental Questions 19
Verification and Validation Procedures 20
Limitations and Assumptions 22
Contributions, Engineering and Scientific Relevance 22
Overview and Organization of this Dissertation 23

REVIEW OF THE LITERATURE 24

Definition of Configuration Design 24
Historical Facts about Configuration Search 24
Different Packing Problems and Relating Algorithms 27
Global Single Objective Optimization Literature Review 40
Multiobjective Optimization Literature Review 49
Summary and Position of the Problem 54
Identification of the Needs 55

A GENERAL CONFIGURATION DESIGN OPTIMIZATION METHOD 56

General Configuration Design Problem Formulation and Complexity Measures 56
Configuration Design Problem Analysis 73
The Research Hypotheses and Their Verification 101
Conclusion 103

METHOD DESCRIPTION 105

The Configuration Design Optimization Method 105
Method Description 105
Implementation 122
Conclusion 130

TEST CASES AND ANALYSIS OF THE RESULTS 131

Validation and Quality Criteria 132
Academic Examples 138

Engineering Applications 162
Conclusion 182

Table of Contents (Continued)

Page

DISCUSSION AND CONCLUSION 185

Introduction 185
Overview of this Dissertation and Achievements 185
Discussion of the Results and Criticism 188
Implications for Future Research 191
Other Implications and Criticisms 192
Summary of this Dissertation 193

APPENDICES 204

Technical Parameters 195
Input Files and Libraries 205

BIBLIOGRAPHY 211

LIST OF TABLES

Table		Page
I.	Classification of the categories of optimization methods.	7
II.	Optimization methods classification.	41
III.	Total variable space size for N objects discrete and continuous CDPs.	81
IV.	Total variable space size for 12 objects discrete and continuous CDPs.	81
V.	Worst case roughness with 12 objects CDPs.	85
VI.	Minimum Dilution of the feasible domain for the 12 objects CDPs.	87
VII.	Number of solutions of the 2D and 3D CDPs in the worst case and the known best cases for N permutable objects.	94
VIII.	Number of solutions of the 2D and 3D CDPs in the worst case and the known best cases.	94
IX.	Number of solutions of the 2D and 3D CDPs in the worst case and the known best cases.	95
X.	Number of solutions of the 2D and 3D CDPs in the worst case and the known best cases.	95
XI.	Number of solutions of the 2D and 3D CDPs in the worst case and the known best cases.	96
XII.	Dilution for N objects.	96
XIII.	Logarithms of the Best dilution (and worst dilution) for 12 objects CDPs.	97
XIV.	Comparison of significant terms of equations (14) and (20).	114
XV.	Number of objects and number of variables for each test case.	132
XVI.	Initial data for the 2D and 3D cube CDP.	140
XVII.	Complexity criteria for each cube test case.	141
XVIII.	Classification of the methods based on the distances from the target extreme planes criteria.	153
XIX.	Classification of the methods based on the number of objective functions and penalty function evaluation.	160
XX.	Final Classification of the methods.	161
XXI.	Satellite CDP components degrees of freedom.	166
XXII.	Results of the Single Objective Optimization Problems for the satellite CDP.	167

List of Tables (Continued)

Table		Page
XXIII.	Results of the multi objective optimization problem for the satellite CDP	169
XXIV.	Car engine CDP components degrees of freedom	174
XXV.	Results of the single objective optimization problems for the car engine CDP	175
XXVI.	Results of the multi objective optimization problem for the car engine CDP	177
A-I.	Evolution of the Trend of function F for different slopes (a) A=10, n=1000	196
A-II.	Evolution of the Average and Standard Deviation of the Number of Generations required for a simple GA to reach convergence calculated over 20 runs for each measure	197
A-III.	Evolution of the Average and Standard Deviation of the precision achieved after convergence in variable space	197
A-IV.	Evolution of the Roughness of a CDP involving ‘a’ objects of category A and ‘s’ static objects	200

LIST OF FIGURES

Figure		Page
1.	The design cycle (Ullman 1993).	2
2.	The product life (Ullman 1993).	2
3.	Part design, assembly design, and part analysis with CATIA V4 (Courtesy of Dassault Systemes, France).	5
4.	Product design and computation.	6
5.	The iterations in design.	8
6.	BMW car engine (Courtesy of Dassault Systemes, France).	11
7.	Equipment of a military airplane.	12
8.	Aerospace component design (Courtesy of Dassault Systemes).	13
9.	Problems classification (Lewis and Papadimitriou 1978).	**¡Error! Marcador no d**
10.	Tangram or Chinese puzzle (1864).	25
11.	Two Dimension Polyminos (Golomb 1970).	26
12.	Three Dimension Polyminos (Golomb, 1970)	26
13.	Packing and cutting problems classification (Dyckhoff 1990).	29
14.	Stock cutting and bin-packing problems.	30
15.	Scheduling problems.	31
16.	Packing squares into a square container.	33
17.	Rectangular stock cutting (Dyckhoff 1990).	34
18.	Trailer loaded with standard pallets.	35
19.	Plant Layout Design.	35
20.	Groemer packing showing the configurations of smallest area (Stewart 1993).	36
21.	Optimum packing shape for minimizing the volume of the convex hull for less than 56 spheres.	37
22.	Genetic algorithm iterative process.	44
23.	Details of Genetic Algorithms Mechanisms.	45
24.	Domain wide characteristics reflected by the Local Optima.	47
25.	Examples of duality gaps with different depths.	51

List of Figures (Continued)

Figure		Page
26.	Example of multiple types of optima in the variable space and in the objective space.	60
27.	Constrained optimization problem landscape.	61
28.	Example of function Roughness (Rg = 11).	65
29.	Trend definition.	66
30.	Trend explanation.	66
31.	Trend of monotonous functions.	67
32.	Trend of non-convex functions.	68
33.	Function F[x] = a*Abs[x - 5] - A*Cos[x - 5] + A is plot for x in [-2nΠ, 2nΠ] and A = 10, n = 10.	72
34.	Evolution of the Trend of Function F when the slope 'a' increases.	72
35.	Evolution of a Simple Genetic Algorithm performance in function of the Trend.	73
36.	Overlapping penalty function.	83
37.	Landscape of a 5 pallet 1D CDP in which 3 pallets are fixed.	85
38.	Evolution of the Feasible Area Dilution.	88
39.	The Mechanical Functional Links (ISO 3956).	89
40.	Example of mechanical functional links.	90
41.	Constraint 'as far as possible'.	91
42.	Complexity classification.	100
43.	Conventional GA population.	106
44.	Sets GA populations.	107
45.	CDOM flow chart.	108
46.	Correspondence between the Mechanical Functional Links and the Displacement Vectors.	109
47.	The one dimension Pallet Loading CDP (CDP 2) with relative positioning vectors. In this case the relative and absolute positioning are the same.	110
48.	The two dimension Square CDP (CDP 1) with absolute positioning vectors.	110
49.	The 2D car engine problem.	111
50.	The 2D car engine problem with relative placement.	112
51.	Evolution of the number of interference.	113

52. In this example the goal is to minimize F1 and F2 116

List of Figures (Continued)

Figure		Page
53.	The 1D rolling pallet example	118
54.	Landscape generation	119
55.	A simple landscape generated by a one Pallet problem	120
56.	Feasible area size and local search	120
57.	Landscape and discretization	121
58.	Objective with penalty function and Trend	122
59.	The STL file format	124
60.	Example of an object used by the CDOM	125
61.	Accessibility	129
62.	Removal path	129
63.	Proposed methods derived from the Iterative Genetic Algorithm with Range Restriction	131
64.	Extreme planes distances	133
65.	Point spreading control	134
66.	Bracketing the area of the Solution Set in 3D	136
67.	Third criterion references	137
68.	In the 2D version of this CDP the cubes can translate along the x and y axis	139
69.	Objectives of the cube CDPs	140
70.	Complexity classification of the cubes CDPs test cases	142
71.	Example of final set distribution and configurations	143
72.	Three different layers of the 4 cube CDP	144
73.	Distance from TEP1 achieved by the four proposed algorithms	146
74.	Comparison of the distances from TEP1 for 2D and 3D test cases	147
75.	Distance from TEP2 achieved by the four proposed algorithms	148
76.	Comparison of the 2D and 3D distances from the target center of gravity	149
77.	Distance from TEP3 achieved by the four proposed algorithms	151
78.	Comparison of the 2D and 3D distances from the target center of gravity	152
79.	Criterion C2 evolution in function of the number of variables	155
80.	Comparison of the 2D and 3D distances from the target center of gravity	156
81.	Distance from the reference plane average and standard deviation	

of the distances from the reference plane. 157

List of Figures (Continued)

Figure		Page
82.	Evolution of the number of objective functions and penalty evaluations	159
83.	The satellite configuration design problem	163
84.	Accessibility of the satellite components	164
85.	The satellite CDP components	165
86.	Example of maximum compactness configuration	168
87.	Example of maximum accessibility configuration	168
88.	Three - objective Pareto set provided by the CDOM	169
89.	2D views of the three objective Pareto set provided by the CDOM	170
90.	Two extreme configurations. At left, the maximum compactness configuration discovered by the CDOM	171
91.	A simplified car engine submitted to the CDOM	172
92.	The car engine CDP components	174
93.	Example of maximum compactness	177
94.	Example of maximum accessibility	177
95.	Three objective Pareto set of the car engine ECDP provided by the CDOM	179
96.	CDOM solutions	180
97.	Maximum compactness configuration (a and b) and maximum accessibility (c and d)	182
98.	Equal scale cut of the complexity classification for $Dil_{fa} = 1/3$	190
99.	Maximum compactness configuration (a and b) and maximum accessibility (c and d)	196
A-1.	The unit Trend of x with respect to x_0 depends on **x** is in zone A or B and of the respective value of F[**x**] with respect to F[$\mathbf{x_0}$]	196
A-2.	Function F[x] = 2*x - 10*Cos[x] +35 on]14 Π, 18 Π [and 2*x on [0, 14Π] and [18Π, 20Π]. Trend[F] = 0.91	199
A-3.	Function F[x] = 2*x - 10*Cos[x] + 45 on]6Π, 10Π [and 2*x on [0, 6Π] and [10Π, 20Π]. Trend[F] = 0.57	199
A-4.	Function F[x] = 2*x - 10*Cos[x] + 55 on]Π, 5Π[and 2*x on [0, 5Π] and [5Π, 20Π]. Trend[F] = 0.41	200
A-5.	Landscape of a 5 pallet 1D CDP	201
A-6.	Landscape of a 2 pallet 1D CDP in which the 2 pallets can move all over the domain and thus can interact, and 3 static pallets	202

CHAPTER I

INTRODUCTION

Some Considerations on Design

To 'design' is 'to create in order to fulfill a goal' (Encyclopedia Britannica 1995). This goal is typically tied with the betterment of life. The first man made designs were surely the tools and weapons used by the prehistoric men. These were at first crude realizations but as early as 40,000 years ago, considerations of efficiency, reuse and maintainability appeared (Knecht 1994). In the early Chinese Empire and Egyptian kingdom, one man who possessed the entire knowledge about physical phenomena could perform mechanical design. Later, this was again possible when Leonardo De Vinci made his wonderful inventions. But already, when Edison began to write his diaries it was obvious that he needed to restrict his discoveries to the fields he was familiar with (Baldwin 1995). As the real and perceived needs of men keep increasing in complexity and the knowledge about our world expands, it is no longer possible for one man alone to synthesize the total amount of knowledge available to mankind. There, emerges the need for collaboration between experts in order to perform design.

Today, the knowledge of these experts alone is no longer sufficient to do design. As the number of elements of a mechanical system and the demand for higher performance increase, as the choice of materials broaden and the understanding of the physical phenomena deepen, engineers have to rely more heavily on complex mathematical models and computers in order to create. Even the methodology used to create must be standardized to be able to give birth to quality products with minimum effort.

This methodology, which is called the Design Process or Design Cycle (Figure 1), starts with the understanding of the need and its translation into specifications. Then, the specifications are translated into concept definitions, which give birth to one or several detailed designs.

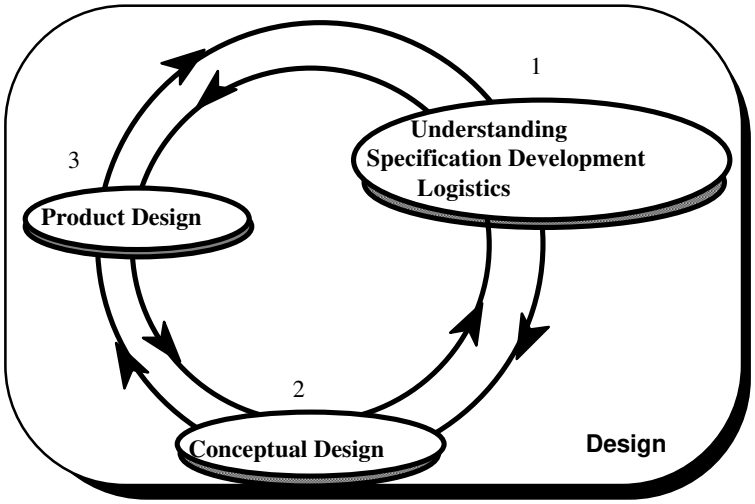

Figure 1. The design cycle (Ullman 1993).

The product is only an embodiment of the design. It goes through different stages, and the sequence of all these stages is called Product Life (Figure 2). The Design stage, which is the context of this work, occurs very early in the product's life and has a tremendous impact on its future. It is during this stage that the product's entire lifespan must be considered in order to foresee and prevent eventual problems. For example, knowing that the product must be disassembled in less than 20 minutes will influence the choice of fasteners and the positioning of the different components of the system being built; knowing that it must be recycled will influence the choice of material of all its components.

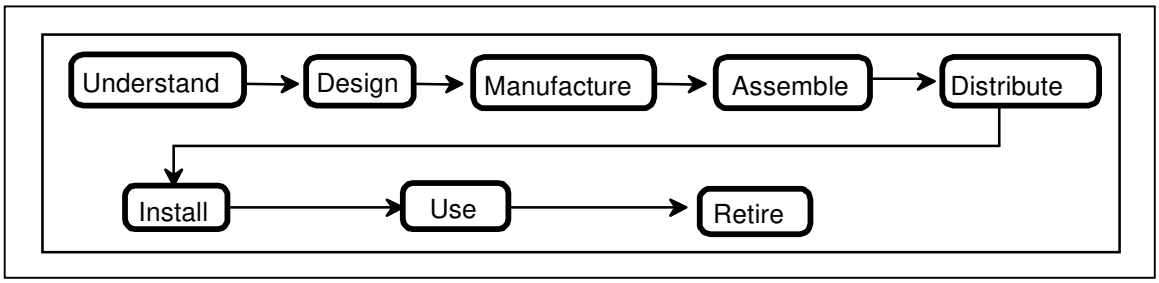

Figure 2. The product life (Ullman 1993).

As the number and technology of system components increase, their design becomes more difficult. It takes many years to design a new airplane and just a few months to build it. While today's engineering techniques are efficient to provide high quality single parts, it seems that many of the difficulties lie in the design of assemblies. Virtually designing assemblies becomes the key of modern product design since first, they represent the final product as it is provided to the final client, and second, for the manufacturer, they represent its final goal.

Reducing the amount of effort, intellectual as well as financial, required to give birth to a good quality product is the leading concern of the research in design.

The Steps of Design

Designing is a process that can be decomposed into 3 interconnected steps, (Ullman 1992, Dixon et al. 1990) (Figure 1) namely specification development, conceptual design and product design. The aim of these steps is to transform the need of the customer into a set of specifications so that the final product can satisfy the user, be manufactured, sold, maintained and recycled competitively.

For this purpose, all that can happen to the product during its life must be taken into account. Usually, this is done by first incorporating the knowledge of past projects into the current one. Second, if a new feature, a new technology or a new need appears during the design process, it can be taken into account by going back to a previous step and modifying either the specification development or the conceptual design. Since the conception of a complex artifact can take several years, it is important to be able to modify the design according to the new technologies, changed constraints and new customer requirements.

Understanding, Specification Development and Logistics

During this step, a need is identified. Its origin can be a market demand for a new product or the birth of a new technology requiring an update of already existing products. In both cases, the satisfaction of the need is formulated as a problem. Can we find a set of parameters (geometry, topology, and materials) for an artifact such that it satisfies the customer? Do we have the means to build the artifact in a competitive way? A bad understanding of the problem leads to wrong designs, delays, high costs and sometime tragic failures (Ullman 1992). Since much is at stake during this first step, the problem must be

carefully formulated and understood and a significant period of time must be reserved only for this purpose. From this, a set of functional requirements and constraints emerge, defining performances, costs and development time.

Conceptual Design

This is the step where the functional requirements defined during the previous step are to be transformed into concepts. The design team usually uses a two-phase procedure. First, designers generate ideas using such techniques as brainstorming, patents search, journal exploration and even interviews. Second, they select, order and refine the solutions. This process ends when it converges toward a small number of feasible designs worth investigating further.

The main tools useful at this point are rough sketches (It is a well known fact that engineers think better with images than with abstract concepts) (Ferguson 1977, Ullman et al. 1990). The design team criticizes each proposed solution and tries to evaluate its feasibility using the knowledge available.

Product Design: The Context of this Work

During this step, the concepts are embodied. Dimensions are set and the details of the design appear in the form of well-defined components (shapes, materials, and connectors).

In the 1980s, the Car and Aerospace industries were the first industrial sectors to rely on Computer Aided Design (CAD) software, to do design. With these tools, engineers are able to represent all the parts of a system as three dimensional mathematical geometric models. They are also able to define full assemblies and full virtual systems as well as the tools and machines necessary to build their components.

Analysis programs, as well as other engineering support software, are usually coupled with CAD software to form a super class called Computer-Aided Engineering (CAE) software. They allow engineers to evaluate the reaction of the 3D geometric representations of the product to external stimuli such as loads, vibrations, and temperatures. Stresses, strains, mode shapes, dynamic properties and other quantities are useful to take decisions about the further evolution of the project (Figure 3).

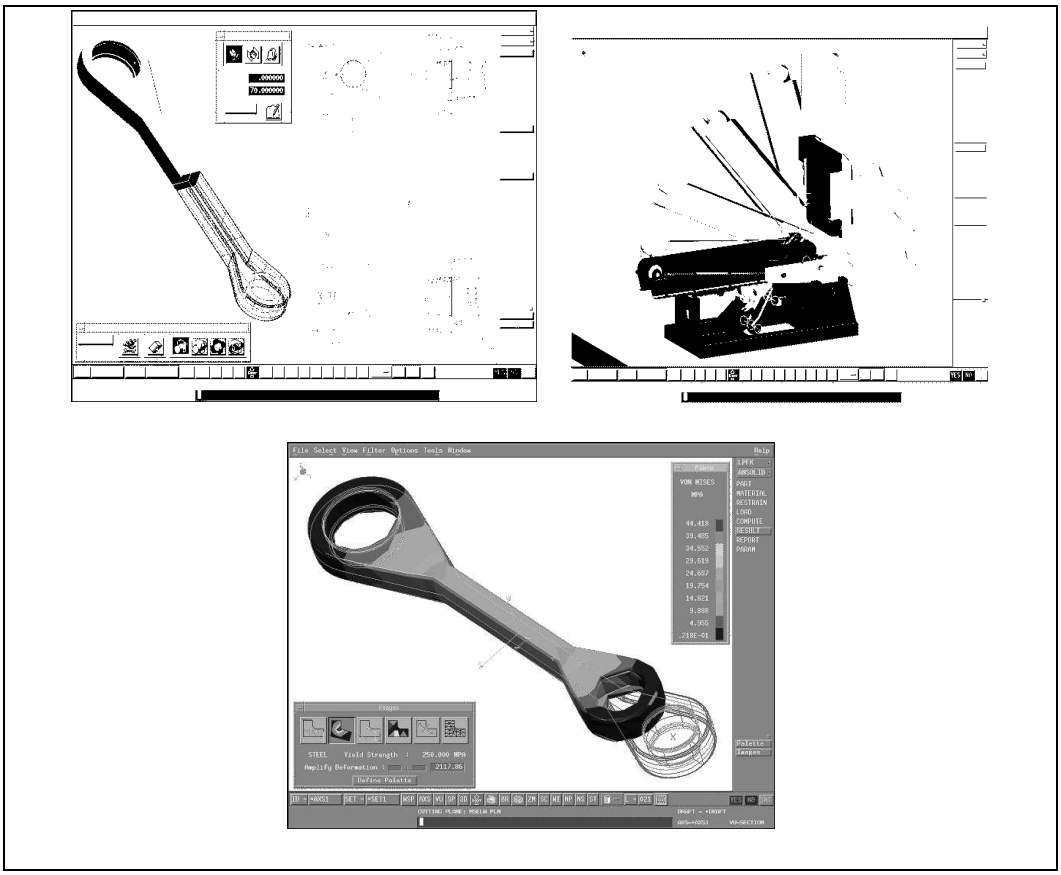

Figure 3. Part design, assembly design, and part analysis with CATIA V4 (Courtesy of Dassault Systemes, France).

The rush toward the Computer Aided Design (CAD) and Computer Aided Engineering (CAE) software is driven by the cost and time reduction that these software bring. They allow insight into the behavior of the product before any mock-up is built. Second, they allow to 'see in 3 dimensions' what the product looks like, which is by far superior to any blue print. Third, they make any modification easier and they reduce the amount of paper work inherent in each project. They ease the information management between people working on the project even when they are not working at the same geographical location. They also ease the management of the resources involved in the product life. Design engineers can refine the digital mock-ups of the product, change parameters and models definition, analyze them during an iterative process until they are satisfied (Figure 4).

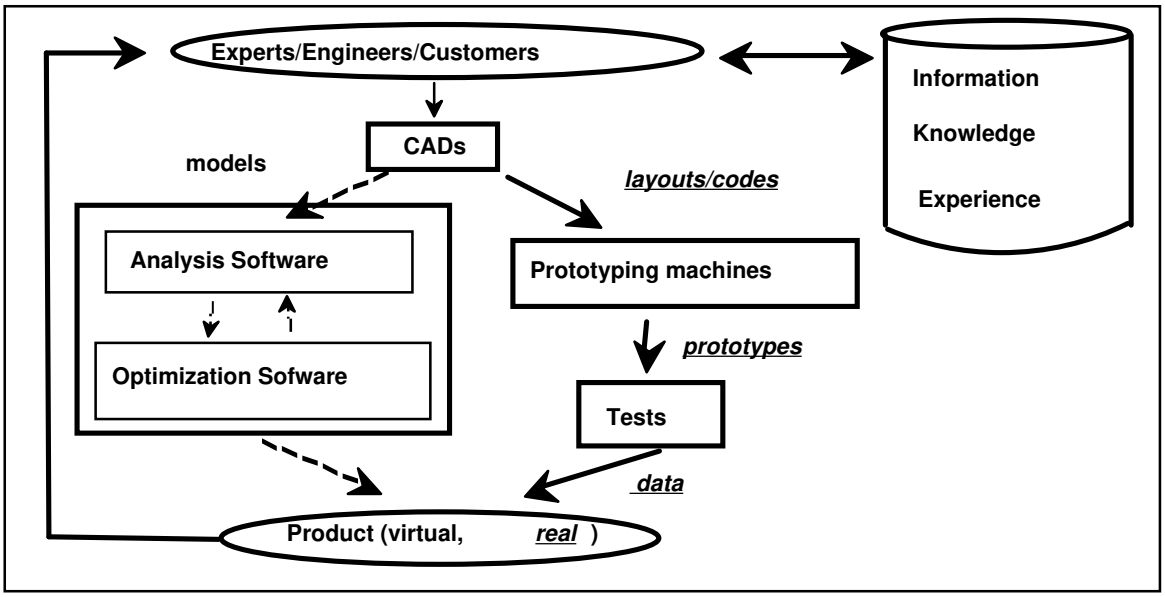

Figure 4. Product design and computation.

Large companies are already claiming that the sum of all these advantages cuts the design time and its cost. Ford, for instance, declared that 3D CAD cuts by 20% the design changes made to fix manufacturing problems and reduces the time to market by 9 to 24 months (New York Times 1997). Matra agrees that 3D technologies cut the study time by 20% and adds that it helps to reduce their mock-ups budget by 30 to 50% (HarVest 1997). In this last domain, Boeing is even going further by giving up the real mock-ups and relying only on digital mock-ups. Their 777 airplane was built without any physical prototype (Rowell A. 1997).

A logical next step after the definition of digital prototypes consists of taking advantage of the quick possibility of analysis and redesign of these prototypes to increase their performances. Optimization is one of the techniques imported from mathematics and made available to engineers thanks to the ability to translate the shapes, behaviors and states of the products into equations. Until now, it has been tightly connected to analysis software alone, which provides the functions to be optimized. The aim is to refine the product dimensions and topology in order to optimize some quantities such as weights, deflections and stress. Among the three types of optimization problems that engineers have to tackle i.e.

shape optimization (changing the shape of one part), topology optimization (changing the connection between the parts) and configuration optimization (changing the location of parts), only the first one is embedded into CAE software. The others either still belong to the academic field or rely on the engineer's experience to be performed (National Research Council 1991, p. 56). The existence of many constraints (like geometrical constraints used to defined the assemblies as well as engineering constraints used to bound the behavior of some design component) and one or several objectives (like weight, deflection, volume, position of the center of gravity, maximum stress among others) in engineering projects makes the study of three categories of constrained optimization methods necessary. They must be used to solve the following three types of problems: Single Objective Optimization (SOO), Multi-Disciplinary Optimization (MDO) and Multi Criteria Optimization (MCO).

Single Objective Optimization has been used in Part Design in connection with Analysis Software in order to modify the shape of a single component.

The necessary collaboration of engineers coming from different fields, especially on high-technology projects fostered the development of Multi Disciplinary Optimization (MDO). MDO methods are the optimization methods that researchers are trying to define whenever several fields of knowledge are involved in the making of the product. Often, the objective function to be optimized is the same for everybody; however, each expert has his own constraints to satisfy.

Finally, designing optimal mechanical systems requires the optimization of several objectives at the same time. A different field of mathematical optimization must be used in order to find a set of parameters, which satisfies the customers and engineers' requirements at best. Very often, these goals are antithetical and improving one objective impairs the others. A compromise must be reached. Multi Criteria Optimization (MCO) deals with finding optimal compromise solutions. Until recent years it has been the tool of choice of Economists and Decision-Makers. It is now spreading, first to manufacturing, which must satisfy both engineering and management constraints, and second to engineering design in order to provide multiple designs satisfying the same specifications (Table I).

Table I. Classification of the categories of optimization methods. These methods are used to tackle the 3 types of mechanical engineering design refinements. The optimization category

can be used: commonly in industry (A), begins to appear in industrial software (B), only for simple problems (C), at research level (D).

	SOO	MDO	MCO
Shape Optimization (Part Design)	A	C	D
Topology Optimization (System Design)	B	C	C
Configuration Optimization (System Design)	C	C	D

Once the technological choices have been made, the study of the design process shows an incessant iterative procedure between the design of the system components and the design of the full system, each one impacting the other. Using the integration and synthesis capabilities of his or her brain, the engineer goes back and forth between Part Design (choosing the shape and the material) and System Design (assembling and choosing the right position and connectors between components, Figure 5).

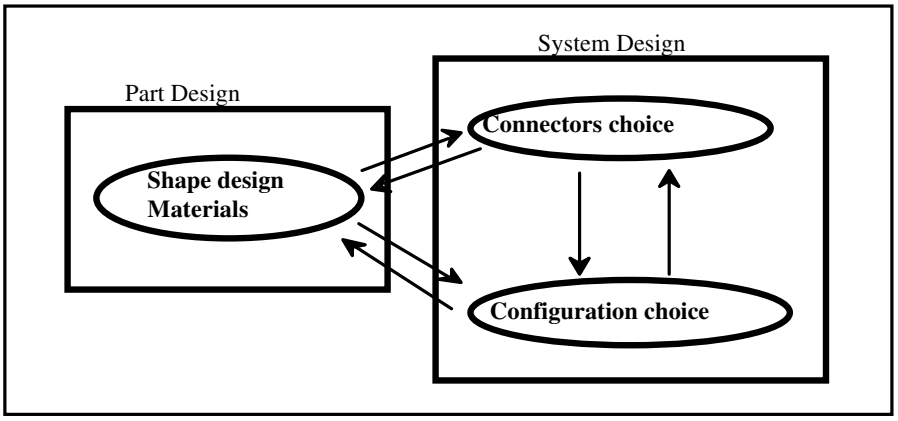

Figure 5. The iterations in design.

When relying on automatic optimization, the integration work done by the engineer and his teammates is by-passed because of the difficulty of simulating abstract thinking on computers. Ideally, system level optimization should be run concurrently with shape design but this is utopian since first, system level optimization is still a young discipline, seldom provided by current software and, second, shape optimization computational cost is too high and thus automatically reshaping an object because of system level optimization results takes too much time. Furthermore, since the components of a system are usually bought from

external independent suppliers, they are seldom redesigned. This is the case for plant or steel structure design for which there is a restricted choice of beams and connectors and most of the work consists in choosing the right ones and then assembling them such that the design satisfies the customer requirements. Relying on this restriction, some software, developed by the academic community, are trying to address the iterations between System Design and Part Design (Dixon et al. 1986, Orelup et al. 1987, Ramachandran et al. 1988) with serious restrictions on their capacity to innovate during the Conceptual Design stage. In industry, the automated configuration choice (i.e. the placement of components) begins to appear in fields such as Ergonomics. With the use of mathematical models of human beings within software such as ADAMS, the different components of cockpits, car or tank interiors, can be placed such that the ease of access and space use are maximized (Rowell A. 1997). These last applications belong to the category of configuration design.

Motivations

This work was first motivated by curiosity about configuration problems in engineering and how they relate to packing games and theoretical results in this domain. Then, the realization that, in industry, assembly type problems were addressed only with rules of thumb and engineers experience gave a strong motivation for developing a method able to address configuration optimization problems at the system level. Eventually, the absence of such type of optimization in CAD software, these emerging tools of choice for engineers and architects, confirmed the choice of the subject of this work. Indeed, although these software tools offer the possibility to define products according to customer requirements, which fostered the use of constraint satisfaction mathematical methods as underlying tools, they lack the possibility to tune the product's dimensions in order to reach an optimal behavior at the system level. This is mainly because thousands of constraints to satisfy is a completely different matter than satisfying these same constraints in addition to optimizing objective functions. Including the possibility of optimal assembly design constitute a challenge that, if met, reduces the gap between pure specific automatic tools and more intelligent tools imitating real design as it is practiced by engineers.

From a more technical point of view, the absence of work about system level optimal design involving several objectives at the same time is a strong motivation for this work. Powerful search methods are emerging from the fields of mathematics and computer science,

yet the link with engineering problems is still tenuous. How far can we push these methods? Are they reliable enough for engineering purposes? These are two of the many questions that come to the mind of many managers and engineers who would like to improve the design of their products. Additional motivations fostering this research have been identified. By only moving components, one may improve an already existing assembly with no need for reengineering. This is a usual practice in industry when either a problem must be fixed at the last moment or when a new product must be put on the market with a low developing cost. Engineers do not have time to redesign the full system. Can we help them improve an existing one? Since any real engineering design problem must meet several goals at the same time, multi objective issues are likely to occur. Volume, balance, and maintainability are among the various objective possibilities that were chosen for this work. The restrictions in terms of objective type, system size and component types are some of the information seeked along this work in order to help engineers to better practice automated design. Although the different objectives usually conflict with each other's, it is believed that there exists a domain of compromised solutions much smaller than the total Design Variable space. Giving the opportunity to navigate through sets of optimal solutions without additional simulations and optimization will widen the range of investigation of engineers. In order to provide the reader with a frame of reference, three industrial applications of configuration design optimization are presented below.

Most of the new cars built today do not present a revolutionary design. One can find the same basic components in nearly all the cars among which are: the motor block, the fan and its motor, the power steering pump and its reservoir, the brake fluid reservoir, the battery, the water tank, the master cylinder, the transmission, the radiator, the distributor, some pulleys and some pipes (Figure 6). The differences are in the presence or absence of one component, in its shape and quality, in the way it is connected to another component and in its location. For example the fan can be an electric fan and require an electric motor with wires connected to the temperature thermostat and to the battery.

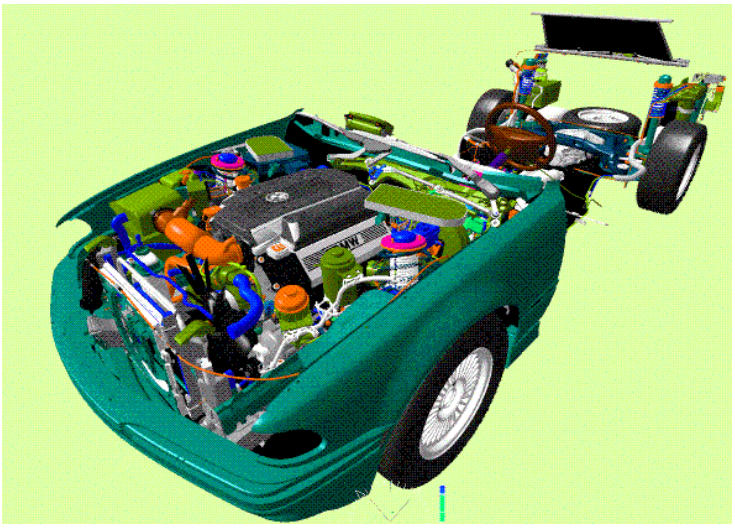

Figure 6. BMW car engine (Courtesy of Dassault Systemes, France).

If it is a conventional fan it will be connected to the motor block by a belt. When going from one option to the other, only some local differences appear, favored by the need for reuse, "carryover and shared parts" dictated by economical pressure, although a completely new design might enhance the performance of the engine (Womack et al. 1991, pp. 110-114). Hence, in order to stay connected to real needs instead of providing a tool that will suggest to restart a full design process each time a small component is changed inside the engine, this work focuses on improving the current engine design by just changing the position of the components, which does not require dramatic changes in the design and manufacturing equipment.

Another example is the avionics problem in which the designers must fit a high number of electronic and mechanical components in a restricted volume (Figure 7). These components must be easily accessible for checking and eventual removal. Some of them have additional positioning constraints since they generate heat, which could damage some other components. Finally, the total wiring distance between the interconnected components must be minimized for cost, reliability, and signal speed considerations. A method that suggests to change components for each different planes will be useless since the electronic suppliers usually propose a catalog of finite number of components. Hence, proposing different locations for the same part in a different context is a balance between non-optimal systems and completely redesigned systems.

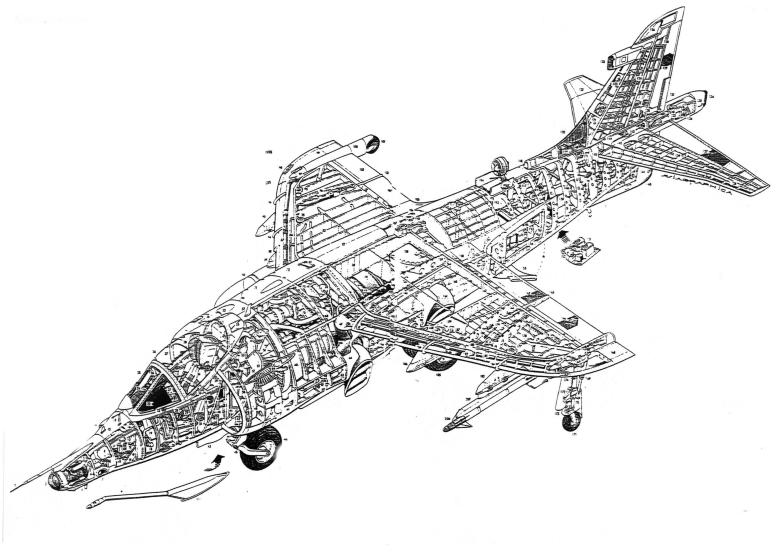

Figure 7. Equipment of a military airplane.

A third example is the design of systems, which are sensitive to the position of their center of gravity. This is the case for payloads of carriers (trucks, airplanes, and boats). Being sure that the center of gravity is at a certain location reduces tire and road wear for trucks, improves the safety when breaking, turning, and reduces the fuel consumption and the driver fatigue. In this case there is no way to redesign the pallets that are standardized (1m x 1.2m or 2m x 2m). Hence, proposing a new pallets distribution is the only way to achieve the goal. This problem increases in magnitude with the design of satellites and rockets components (Figure 8) which are so sensitive to center of gravity location that any error might affect their good functioning and cost hundreds of millions of dollars.

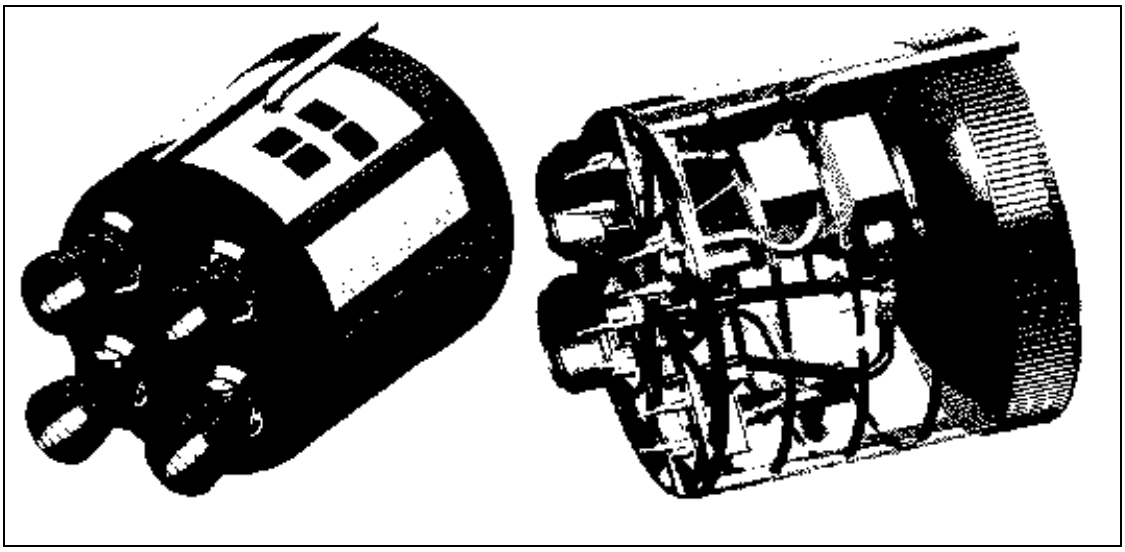

Figure 8. Aerospace component design (Courtesy of Dassault Systemes).

It is important to understand that all these considerations about the placement of objects in space in order to build an optimal system were not the main preoccupation of the engineers as long as major improvements came from other sources (material improvement, new concepts). However, in many engineering fields, revolutionary designs are becoming rare and still improvement is needed in order to stay competitive.

From the scientific point of view, this research was fostered by the absence of general conclusions about the whole family of packing problems in engineering, and which are usually formulated as combinatorial mathematical problems applied to components, which have few common characteristics with real components. Most of the research efforts in this domain were focused on producing ever more efficient algorithms to address a specific problem and thus, which perform poorly as soon as the components changed. Moreover, using different methods to study a category of problems prevents one from drawing unified conclusions about the difficulty of the task. On the other hand, a more general method, even if less efficient for specific problems, is more widely applicable and thus allows comparisons to be made and conclusions to be drawn. Some recent advances obtained in global optimization might be helpful to investigate engineering packing problems. Especially since these problems are Multi Criteria. It is of interest to establish a classification according to explicit parameters involving the objects and the whole system.

Goal of this Dissertation

The aim of the current research is to investigate the field which is the most unexplored in this landscape: system level multicriteria optimization of mechanical systems. It is proposed to define a method, namely Configuration Design Optimization Method (CDOM) which will help to perform system level optimization, to improve the principal values of mechanical systems i.e. their performance, their safety, their cost and the pleasure they bring to customers, by only relocating their components. This way of performing configuration design is complementary with previous approaches in which people were dealing with either the shape of the objects like in (Chapman et al. 1994, Kane et al. 1995), or the way these components are chosen and connected together (Carlson and Pegg 1995; Sthanusubramonian 1992). The proposed approach will also enhance the capabilities of the current Computer Aided Design environments that already allow engineers to assemble 3D geometric models and to optimize each of the components thanks to shape optimization. Keeping in mind that Configuration Design is only a step of the Product Design Process which must stay open for a constant dialog with Part Design (Figure 5), CDOM must propose entries for eventual connections with either other automatic methods or design engineers themselves.

In order to rate the performance of the CDOM, it is proposed to build a classification of the CDPs based on criteria applicable to any of them. Then, a comparison between this classification and the efficiency of CDOM will validate or invalidate the classification.

In addition to the construction method itself, and to the gathering of the solutions to the obstacles encountered to reach the solution, this work will try to answer some fundamental questions about the complexity of Configuration Design and about the expected efficiency of today's algorithms to solve such a problem.

The Fundamental Bases

The study of how one can assemble objects in order to reach a goal falls into a category of problems that require two fundamental changes in the way we, as occidental people, reason in order to solve them. These two changes are first, the recourse to system science -which means abandoning the Cartesian way of thinking relying on decomposition-, and second, the recourse to non-deterministic processes.

In 1637, the French mathematician R. Descartes, in "Le Discours de la methode," built the foundations of the occidental scientific way of reasoning with four fundamental precepts:

- Never take something for true, unless you know it is. Avoid rush and prevention.
- Divide the problems into as many sub-problems as needed in order to better solve them.
- Solve such subdivided problems beginning from the simplest one to the most complicated.
- Count everything in order not to forget anything.

The implicit assumption made by Descartes when he laid these principles was that a problem was decomposable into sub-problems. Unfortunately, configuration problems are not decomposable. One cannot solve a packing problem by taking the objects one after the other and assembling them in such a way that at each step the volume is minimized. This strategy is doomed to fail except for some very special cases. Hence, all the components must be considered at the same time. One might even try to solve the problem for two components, then for three and so on, each time trying to learn and detect a pattern in the solutions evolution in order to build a general rule that could be applied for a higher number of objects. As we will see in the literature review, this is impossible since the arrangement of objects which satisfies the goal (the system solution) can unexpectedly change shape each time an object is added. The mathematical notion that translates this characteristic is non-linearity i.e. the principle of superposition described by Nicolis as "the observed effects are linked to the underlying causes by a set of laws reducing for all practical purposes to a simple proportionality" (Nicolis 1995) does not hold anymore. In practice, we will see, for example, that even a tiny change in the components of a car engine might dramatically affect the final solution. Hence, configuration problems involve two types of non-linearity; first a non-linearity due to the goal; second a non-linearity when the problem is scaled.

During the years following World War Two, a new trend of thought appeared in academic fields and research laboratories of large companies: the concept of a system. Among the famous names of this new trend are Norbert Weiner, at the time professor at MIT and author of "Cybernetics" (Weiner 1948), Weaver Shanon, famous telecommunication engineer who wrote "A mathematical theory of communication" (Shanon 1948). Both recalled the ideas of Ludwig Von Bertalanffy, professor at the university of Alberta (Canada) who

wrote his “General System Theory” (Bertalanffy 1874), and improved that system theory. They defined a system as “ A global unit made of components which interact dynamically with each others and are arranged in function of a goal“. This system might exhibit complex behaviors i.e. changing one of its parameters can dramatically change its characteristics and even destabilize its dynamic behavior. Hence, studying the system by decomposition cannot be done. This trend of thought is well known in the mathematical aspects of constraint satisfaction in design: it is better, from a theoretical point of view, to use variational design - in which all the equations of the design are considered all together- rather than performing history-based or parametric design in which groups of equations are solved sequentially. The main drawback of the first method is the size of the problem and the absence of guaranty to find a solution because of the non-linearity of the problem, while the second method is less impacted by the size of the problem but has even less potential to find a solution.

However, in our misfortune we are lucky, since among the various existing complex systems (chemicals, biological, social) this work deals with the simplest: mechanical systems.

Thus, a first characteristic of a CDP is its inability to be decomposed. Let’s go on and try to identify other characteristics in order to choose a methodology to solve it. Unlike many mechanical engineering problems, which are solved using the integration of differential equations over more or less regular domains, CDPs are not formulated with Partial Differential Equations. At first glance CDPs seem closer to packing problems rather than to many conventional engineering and physics problems. The intrinsic nature of the packing problem made them fall into the domains investigated in the field of combinatorial geometry. Some well known packing problems involve objects of simple shapes like squares, rectangles, circles or spheres because their volume and boundaries are easily defined analytically. This allows mathematicians to study examples, generalize them and try to build theories.

In discrete mathematics and computer science, problems can be classified according to their Complexity. In these fields, Complexity is defined by 'the minimal cost among all algorithms that solve the problem'. It is convenient to define the 'cost' as the time required to reach a solution. This time can be related to the size of the input of the problem. For example, finding the shortest path going from one vertex to another vertex on a weighted graph is a problem whose complexity is dependent on the number of vertices and connections between those vertices in the graph. It is intuitively understandable that the more links and

nodes, the longer it takes to solve the problem. As the dimension of the input grows, the time necessary to solve the problem grows at a certain rate. This rate is the measure of the complexity and the basis on which is built a problem classification (Figure 9). If the complexity grows like a polynomial function of the size of the input (or less quickly like a logarithm) the problem is said to belong to 'P'; 'P' standing for 'the set of all the problems which can be solved by a deterministic algorithm in polynomial time' (Sedgewick 1983). For example, the complexity of sorting problems grows like the logarithm of the number of objects to be sorted. These problems belong to 'P' and are considered easy to solve.

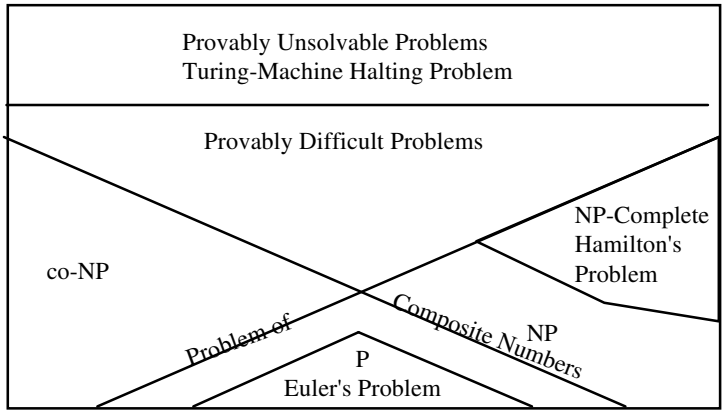

Figure 9. Problems classification (Lewis and Papadimitriou 1978).

Aside from this first category, mathematicians have found some problems whose complexity grows exponentially in function of the size of their input. These problems are usually separated into two categories: those having a solution and those having no solution. It is easy to understand the distinction between those. If one wants to prove that a problem has a solution, one needs to exhibit at least one of its solution. On one hand, if the problem has one solution then, maybe by chance or by ingenuity, this solution could be very quick to find. On the other hand, if the problem has no solution, the only proof of this fact consists in a thorough investigation of all the cases. This later process is guaranteed to take much longer than the former. Mathematicians put in the set 'NP' all the problems having a solution but requiring more than a polynomial time to be solved. These problems are said intractable

(Garey and Jonshon 1979). The status of this set is ambiguous since nobody knows if one day a polynomial time algorithm will not be discovered for one of its members. The 'Provably Difficult Problems' is a set composed of problems that always require an exponential time to be solved. Finally some problems are unsolvable. There will never be a method to solve them (Lewis and Papadimitriou 1978).

Packing problems, as well as Discrete Configuration Design Problems, belong to the NP set since there is always a minimum volume in which the objects can be packed, and since the time required to find the best configuration using conventional deterministic optimization methods grows exponentially with the number of objects. It is then recommended to use non-deterministic algorithms to solve them (Lewis and Papadimitriou 1978). The average case time complexity of conventional heuristics to solve NP complete problems is dependent on the type of input data. There usually always exists a problem that the method is not able to tackle during the average time predicted. On the other hand, the non-deterministic techniques can perform poorly on any data, however this has very little chance to occur (O'Rourke 1993, p. 318). For an non experimented observer, the result is the same: in some cases the method fails to give a good answer because of lack of time. A second thought of the difference between these two methods shows that running a second time a non-deterministic method on such a problem gives an additional chance to discover the solution but once a deterministic heuristic has failed for a problem it will always fail with the same input.

However, Engineering CDPs (ECDPs) are generally not discrete and thus the previous complexity measures calculated as a function of the input length might not be applicable although Continuous CDPs can be considered as an extension of Discrete CDPs. Complexity measures of continuous functions, often called epsilon complexity (ε-complexity), were already extensively studied in the general case (Ko 1991) based on the idea that precision must replace input length (Ko 1991, Packel 1987). However, strong assumptions on the regularity of the functions are usually needed to get useful complexity bounds. These assumptions are difficult to guarantee for CDPs. More over, these methods do not indicate how to evaluate the complexity of mixed discrete/continuous functions. A study of time-based continuous phenomena complexity was proposed by Nicolis (1995) in his book "Non-

Linear Science", nevertheless the functions studied in this dissertation do not exhibit such level of non-linearity as the chaotic phenomena described in his work.

Eventually, ECDPs differ from their packing siblings by another characteristic: They are multi-criteria in nature. Thus, in addition to volume objective other goals must be taken into account such as center of gravity, maintainability and other relevant characteristics of mechanical systems that engineers consider when building an artifact. In order to achieve an optimal design, one must now add Multi-Criteria Optimization (MCO) to the collection of tools needed to solve the problem. It will be shown that this additional characteristic brings many restrictions to the method that might be applied.

Thus, an Engineering Configuration Design Problem consists in placing components in order to build a system such that its characteristics are as close as possible to target values. They constitute a super set of all the discrete packing problems and inherit one of the latter well known characteristic i.e. they cannot be decomposed. Their differences lie in the fact that they involve continuous variables, and several goals. However in spite of these differences they are strong evidences for the use of non-deterministic algorithms to tackle the ECDPs.

Fundamental Questions

The nature of the Engineering Configuration Design Problems (ECDPs) orients the choice of the search method toward non-deterministic algorithms such as Genetic Algorithm, Simulated Annealing or Restarted Local Search. In spite of this choice, much is still to be explored, discovered and explained. Hence, aside from the definition of the method itself, the author will attempt to take advantage of its application to various problems, ranging from pure mathematical functions to engineering oriented cases, to answer the few fundamental questions pertaining to the fields of engineering configuration design and optimization. These questions respectively concern:

- The complexity of Configuration Design Problems.
- The type of formulation (Objectives and Constraints).
- The capacity of adaptation of a search.
- The possibility to provide multiple solutions.

Given a set of freeform objects to place in space, and given some objective functions to be optimized in function of the position of these objects, the answers to the following questions are sought.

1. Can the complexity of the ECDPs be characterized?

    1.1. What is the correlation between these characteristics and the performance of the CDOM?

    1.2. What is the relation between the morphological mechanical characteristics (component size and shape, system size) and the complexity of an ECDP?

2. What are the factors that can decrease the complexity of an ECDP?

    2.1. It was already shown that even if the intractability of NP-complete packing problems cannot be broken, we can, by placing the components relatively to each others, increase the size of the manageable problems such that the number of objects involved is useful for common industrial applications. This was done for very special cases. Can we adapt these implementation to ECDPs?

    2.2. Heuristics are not only defining new problem formulations but also new combinations of optimization methods to tackle the packing problems. How does this help solving the ECDPs?

3. Searching for something in an unknown environment requires a balance between imagination (exploring a new path), and deduction (exploiting gathered information i.e. drawing conclusion from clues). In the case of the ECDPs, what could be the gain of an adaptive method? What could be its drawbacks?

4. CDPs usually have multiple solutions either because the only objective of the problem has multiple global extrema or because the CDP involves multiple objectives. In both cases it is interesting to have a sampling of the different solutions. How can the CDOM provide multiple solutions to configuration problems?

    4.1. Most of the numerical mathematical algorithms are built on the need for convergence toward one solution, in this context what must be done in CDOM in order to find and maintain a sampling of different solutions at the same time?

    4.2. How does this new constraint affect the choice of algorithms to solve an ECDP?

Verification and Validation Procedures

Procedure

First, a classification of the well known CDPs will be done based on natural mechanical characteristics that will be translated into mathematical measures in order to rate the complexity of the ECDPs. These characteristics will come from a study of the packing problems encountered in the literature review. This will contribute to answer question 1.1. From there, some improvements will be suggested in order to decrease the complexity of the

ECDPs by other means than decreasing the number of components involved in the problem (2.1) and they will be implemented in the CDOM.

Second, the CDOM is applied to a set of increasingly difficult configuration design problems with known solutions. Statistics about the rate of convergence and the quality of the solution obtained will provide the information necessary to compare the different formulations and strategies implemented to answer the second question.

A comparison between the classification proposed and the results obtained with the CDOM will provide further material for demonstrating the conclusions about questions one and two.

The CDOM will be tested again with different strategies favoring exploration versus learning and the reverse. A proposition for adapting the parameters of the method to the difficulty of the problem will be made and the test cases will be run again. The results thus obtained will be compared to the results obtained to answer question one and two in order to detect some improvement. This study will answer question 3.

The CDOM is intended to provide multiple solutions to the ECDP. The quality of the results obtained for the first three questions as well as the way the CDOM is built will answer question 4.

Measures of Quality

Since the exact answer to any real complex configuration problem is rarely, if ever known, the performance of the CDOM will be tested against customized test cases. The measures of quality chosen to compare the performance are the following :

- Quality of the results : The quality of the results will be expressed as a distance between the result given by the method and the exact answer expected whenever this information exists. If the best solution is not known, a relative classification will be done. For this purpose, the distance must be carefully chosen in order to be meaningful when comparing different test cases.

- Quality of the method :

  - Speed : This is a relative measure of one method with respect to the others. It will be expressed by the number of objective function calculations.

  - Repeatability : This property of the CDOM translates its ability to give the same result quality when reapplied to the same CDP.

  - Generality : The method is said to be general if it could be applied to a majority of test cases without a change, and gives satisfactory results.

Limitations and Assumptions

The following limitations and assumptions are valid for all this work. The author assumes the existence of an area of the Design Variables Space allowing the search for an improvement of the system performance without changing neither the shape nor the connectivity of its components. There always are some degrees of freedom in the system i.e. this work will not study over or perfectly constrained systems. The problem has always a solution because the system usually exists and thus either its components are already at their optimal position or, given the previous hypothesis, they can move in order to improve some performance criterion. The formulation of the problem as an optimization problem with positioning constraints is a relative problem.

Since this work deals with mechanical systems, we will assume that the spaces of Design Variables are not only bounded but also finite because of the finite accuracy required by most manufacturing equipment. Thus, the cardinal (or measure) of the Design Space, feasible and unfeasible regions together, is always known. However, it is assumed that the use of operators such as gradients will be computable and the methods using such operators will continue to be applicable.

The aim of this work is neither to study the various objective functions and their degree of non-linearity nor to compare the mathematical models of object descriptions. The main point of interest is the method itself. Thus, the simplest models are used for describing the objects keeping in mind what this simplification means for the validity of the results. A criticism of this simplification is included in the discussion of the results. The main focus is on the method when applied to configuration problems and on the characteristics of configuration problems.

Contributions, Engineering and Scientific Relevance

The main contribution of this work to the engineering field is the Configuration Design Optimization Method that allows improvement of some characteristics of a system by moving its components. This method can handle multiple objectives at the same time. From the point of view of configuration design, this work targets a better understanding of the impact of the system size on one hand, and on the other hand, components size, shape and number, on the difficulty to find a good solution to the ECDP.

In the field of computational mathematics, this method is an attempt to establish the bounds in the performance of advanced non-deterministic strategies when applied to real life problems. These bounds must be easily expressed using, for example, the number of objects, their size, shape and some characteristics of the system in which they are evolving.

Beyond the engineering and optimization goal, this work is an attempt to identify which kind of information might break the intractability of Engineering Configuration Design Problems and how to introduce it into an automated searching process.

The scientific relevance of this work consists in showing how one can improve the performance of a search relying on this incomplete information, to reveal the limitation of such strategies, and how, when applied to real problems like configuration design, for which the unformulated knowledge is high, the missing information can come from the global understanding of the engineer.

Overview and Organization of this Dissertation

This dissertation is divided into six chapters.

- Chapter I is the introduction which presents the engineering and mathematical context of this work. It also gives a general overview of engineering design and explains some terminology used in this dissertation, its goal and its limitations.
- In Chapter II the literature pertaining to the field of configuration design, and optimization is reviewed. Some historical issues concerning packing problems are presented as well as the various heuristics used to solve them.
- In Chapter III a general formulation for all the CDPs is proposed. Based on this formulation several complexity measures are presented and applied to several well-known packing problems. The research hypotheses are given.
- In Chapter IV the reader will find a presentation of the CDOM and of its implementation. Each step of the method is detailed as well as the objective and constraint functions. Several strategies based on the conclusions reached in Chapter III are used for validating the research hypotheses.
- In Chapter V the reader will find the testing procedure, the test cases with their objective functions and quality criteria, as well as their associated results. Comparisons are made and technical conclusions are drawn about the CDOM and the CDPs. Two engineering cases are then submitted to the best method identified by the analysis.
- Chapter VI is the general discussion about this work. A criticism of the method concerning its usefulness for industrial size assemblies is presented. Modifications are proposed for compensating the weakness identified in Chapter IV and further areas of research are identified.

CHAPTER II

REVIEW OF THE LITERATURE

Introduction

No literature deals directly with the topic of this work i.e. Optimal System Configuration Design through the placement of 3D freeform objects. Hence, a series of related topics is reviewed. First, the definition of configuration is restated and some common configuration search problems are presented. Since the placement of 3D objects is first a combinatorial problem, the usual packing problems are reviewed as well as their main results. Extending these studies to freeform objects requires a migration from discrete to continuous variables and thus changes the type of optimization used. The fact that Configuration Problems are non-linear with many local optima dictates the focus on global non-linear optimization techniques.

Second, since the Engineering Configuration Design Problems concern the optimization of multiple criteria, multiobjective optimization techniques are reviewed.

Definition of Configuration Design

The Encyclopedia Britannica defines a configuration as a relative arrangement of parts and elements (Encyclopedia Britannica 1995). In the context of design, it refers to the position of all the components of a system with respect to each other's. The way these components are connected is called system topology or system connectivity and their geometrical definition is called shape. Hence, in all the following, performing configuration design means placing the components without changing neither their shape nor their connectivity.

Historical Facts about Configuration Search

The search for spatial configuration of objects is coupled with the desire to achieve several types of goals. Human beings are first trained and then entertained with a multitude of games involving the search for 'good' configurations. In some of these games like Tetris, Tangram (Encyclopedia Britannica 1995)(Figure 10), puzzles or Polyminos (Golomb 1970)(Figure 11 and 12), the aim is very well defined and easily quantifiable. For example,

in the Chinese game called Tangram, the goal is to match an arrangement of seven elementary objects given the contour of the arrangement and the seven elementary objects in disorder (Figure 10). The goal is perfectly clear and one can answer by yes or no to the question 'does the arrangement found match the contour prescribed?'. The positions of the objects with respect to each other belong to a discrete set (translations: {top, bottom, right, left}, rotations: {0, 30, 45, 60, 90,.., 340}). In some other games like legos, the goal could be subjective. One can build a beautiful house in legos and still the answer to the question 'Is it beautiful?' might well be unprecise and thus difficult to quantify by clear discrete answers.

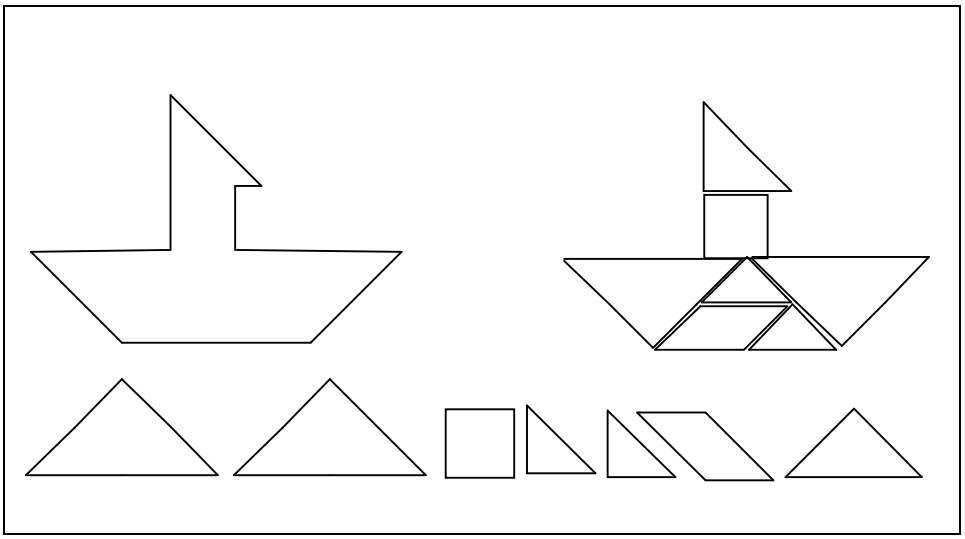

Figure 10. Tangram or Chinese puzzle (1864).

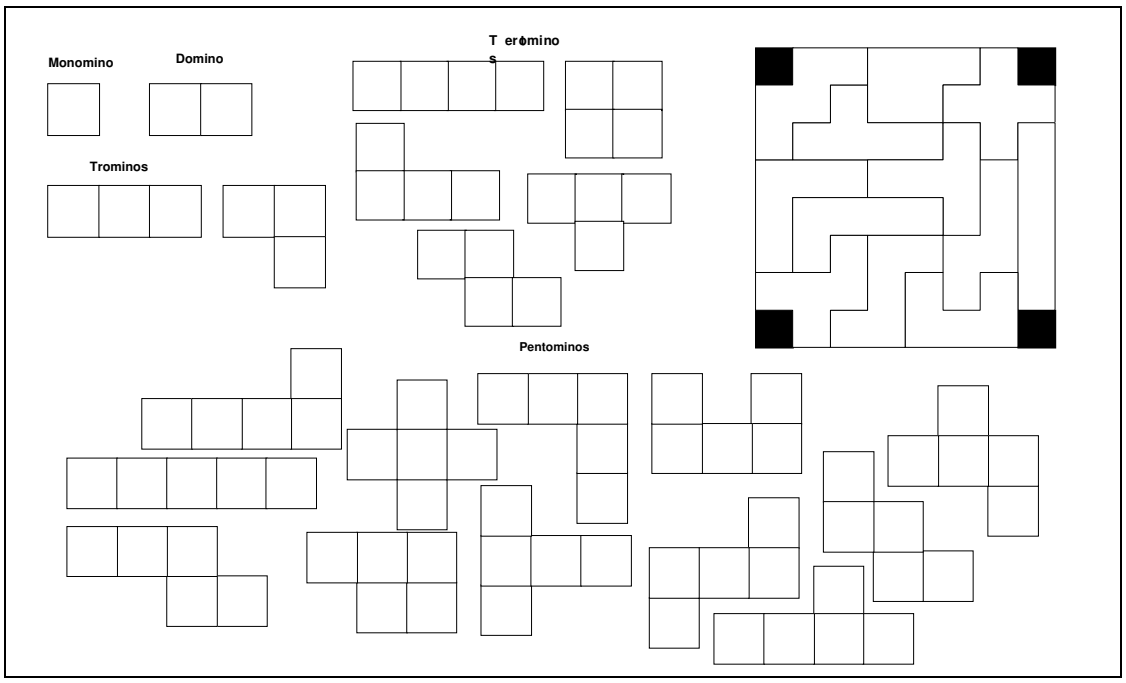

Figure 11. Two Dimension Polyminos (Golomb 1970).

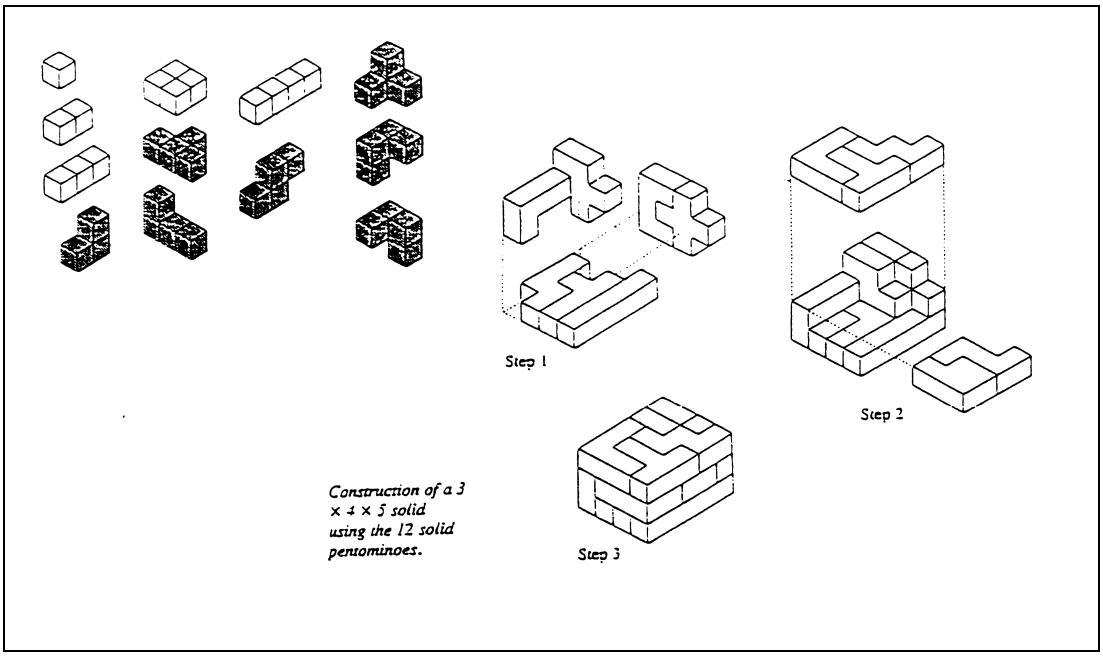

Figure 12. Three Dimension Polyminos (Golomb, 1970).

However, from the theoretical point of view, those differences do not bring any change in the way the problem is solved i.e. using optimization techniques adapted to discrete problems. This is because the number of interesting component positions is discrete and bounded. Some industrial type problems, once formulated, match these same characteristics. For example, loading a truck, a plane, a boat in order to maximize the payload is an activity comparable to Tangram or puzzles and thus is usually solved using combinatorial methods. Placing the same components in order to optimize the position of the load center of gravity falls into a different category of problems since the variables are continuous. Placing the components of the engine of a car, or the chips on a computer circuit board in order to minimize the space loss requires optimization techniques for continuous functions. Finally, placing the components of an airplane engine in order to make it serviceable in minimal time is a much more subjective issue. The biggest difficulty of this type of configuration management lies in the quantification of the value of a design. But once this difficulty is removed, this problem is similar to the car engine configuration design.

Hence, a first glimpse at the literature already shows two broad categories of configuration problems defined by the type of their variables. According to these categories, two different areas of mathematics are used to deal with these problems: combinatorial optimization concerns discrete variables and non-linear optimization for continuous functions concerns continuous variables.

Different Packing Problems and Relating Algorithms

With the increasing power of the search methods, computationally expensive techniques used to solve configuration problems are receiving more and more attention. Common problems involve length, surface or volume minimization, accessibility and maintainability optimization, and balanced loading. One of the most well covered research about spatial arrangement of objects found in the scientific literature is the field of packing problems. They constitute an interesting assortment of theoretical and applied research. The positioning in space of simple components in order to reach a goal is a mental exercise, an entertainment, which ranges from the easy puzzles to still unsolved problems. It has also the characteristics of being appealing for mathematical research because of the simplicity of the problem formulation and the possibility of finding proofs of the results. Their most interesting aspect is the fact that they belong to the family of NP problems. Typical

examples of such exercises involve squares, circles, cubes and spheres packing. Hence, it is believed in this work that studying what was done in this area is a good starting point for understanding the fundamental difficulties of packing in general and then placement problems specifically in order to optimize other functions than volume, keeping in mind the differences emphasized in the previous paragraph.

Packing as Configuration Optimization for Volume, Surface and Length Minimization: A Trend Toward Non-Determinist Search Methods.

Although configuration design involves broader issues than only optimizing the volume, the features of packing problems provide a good sampling of the difficulties that one can expect when dealing with such topics. Moreover, some continuous configuration design problems can be reduced to smaller continuous problems coupled with a combinatorial problem less expensive to investigate (Grignon and Fadel 1996).

Packing problems have been one of the most exciting pastimes for centuries. Puzzles, Tangram, Pentominos and other Polyminos reflect this excitement through Human history. Nowadays, these problems find application in industry. Loading ships, trucks and trains, designing plants, electronic boards, placing mechanical components, cutting stocks, scheduling, are some of the numerous problems that can be classified as packing optimization problems (Figure 13). Dyckhoff (1990) proposed a classification of packing problems based on four criteria: the dimension (1,2,3 or N), the assignment (the type of objective), the assortment of containers and the assortment of objects to be packed.

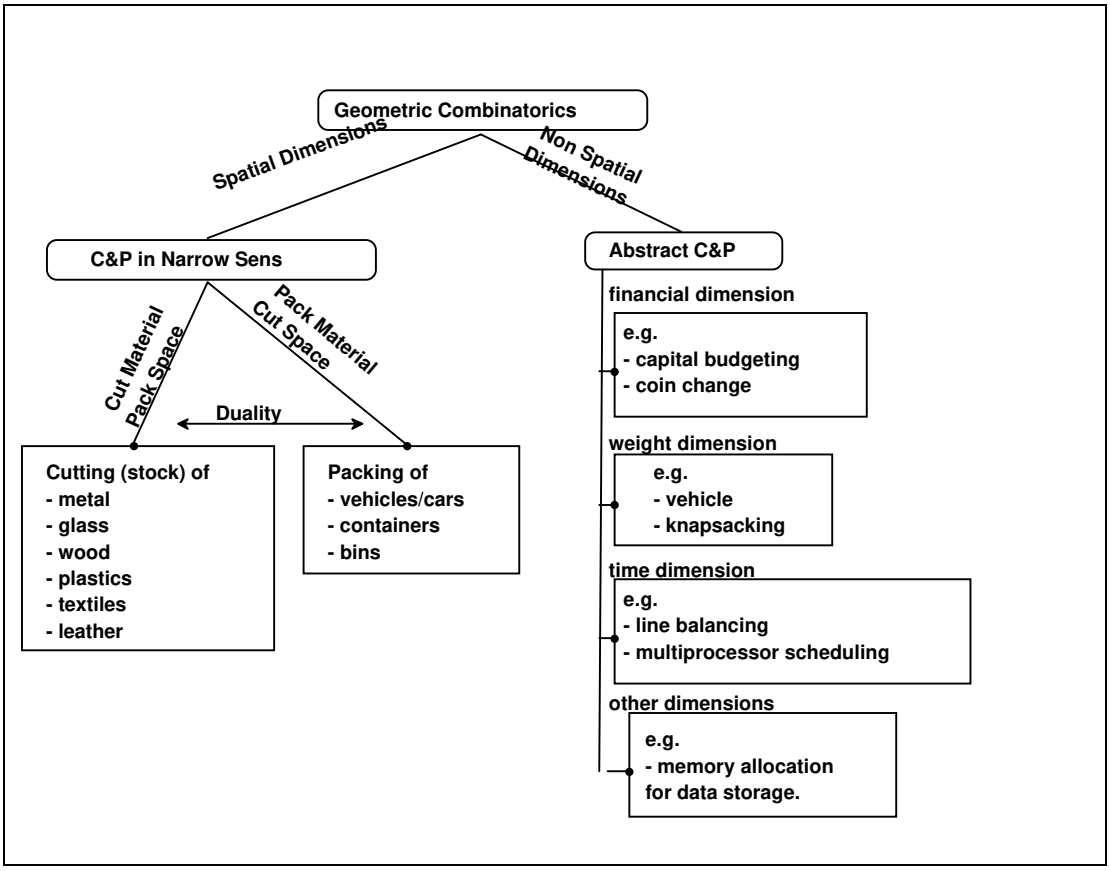

Figure 13. Packing and cutting problems classification (Dyckhoff 1990).

Among geometrical packing and covering cases, those involving spheres, squares and rectangles are the best known because they are the easiest to formulate and thus the most studied. Unfortunately, even these simple cases are known to be difficult problems in the mathematical sense. So far, nobody has yet produced a global formal solution for them and even some mathematicians believe that this event will never happen (Lewis and Papadimitriou 1978). Today's alternative is to accept a nearly optimal solution using computers and heuristics. These heuristics are case dependent and those guaranteeing good solutions are subject to combinatorial explosion. The problem becomes even more complex when the objects or/and the containers are freeform, and/or three dimensional, such that only few recent works exist on this subject.

Bin Packing

The basic problem in bin packing is to find the best arrangement of 1 dimensional objects into 1 dimensional containers in order to minimize the space loss.

Two types of problems are encountered in this category and countless variants exist. First, problems for which the size of the container is fixed and the number of containers must be minimized when fitting the objects within. Second, problems for which the number of containers is fixed but their size is unknown and must be minimized. Stock cutting, bin packing (Figure 14), belong to the first category. Scheduling belongs to the second category (Figure 15).

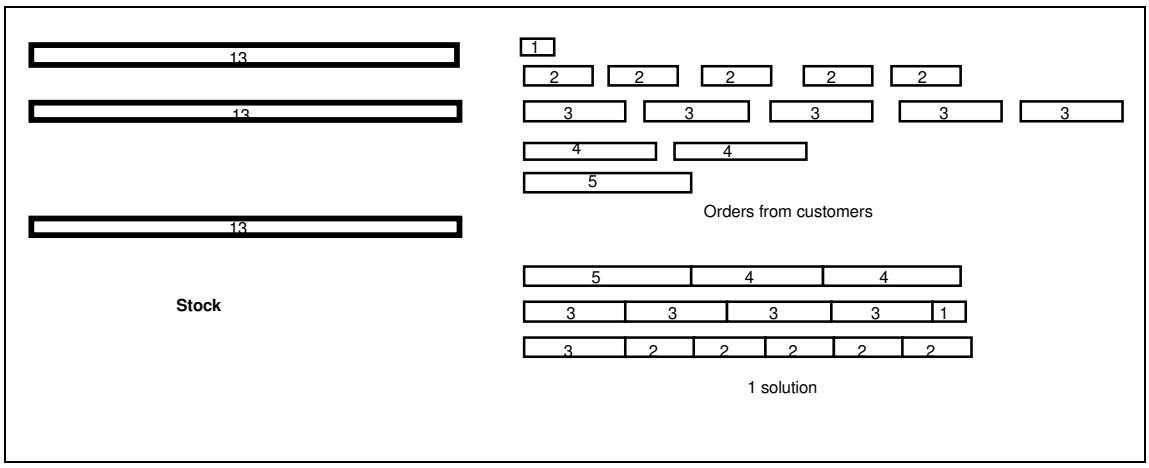

Figure 14. Stock cutting and bin-packing problems.

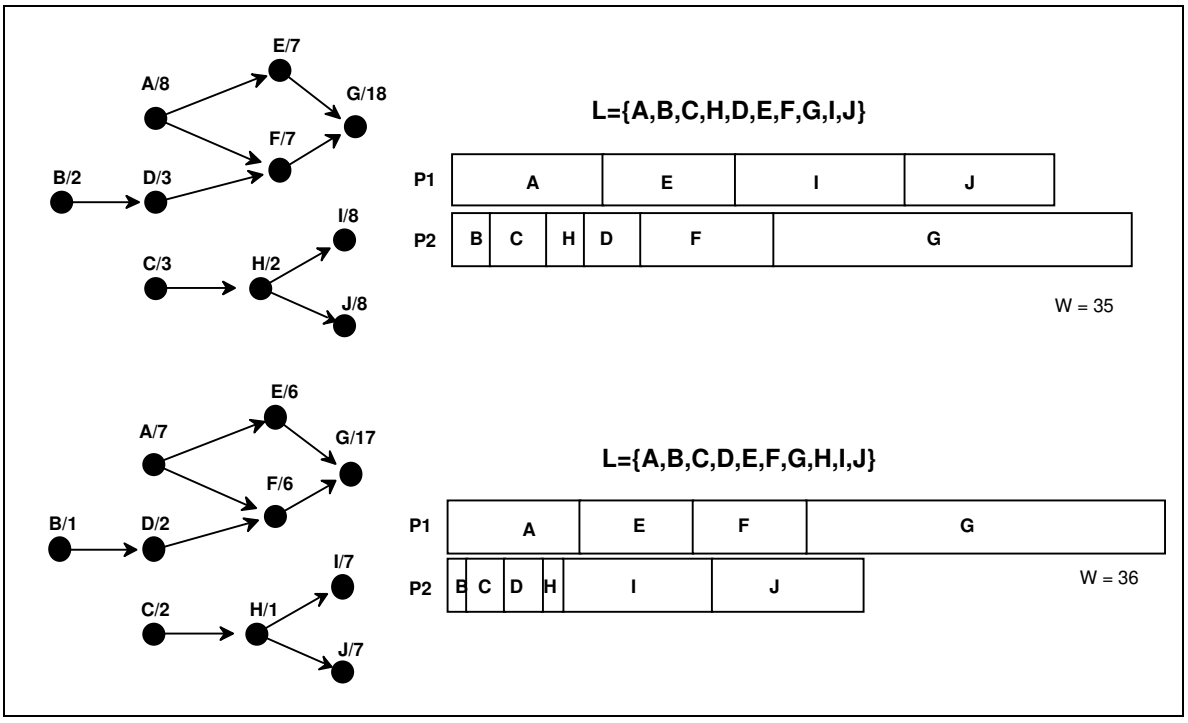

Figure 15. Scheduling problems. These problems are very sensitive to a small change of their initial inputs, which makes difficult predictions of their solution. Even a decrease of the tasks time length increases the final completion time of this two-operator problem (Graham 1978).

Three common heuristics are used to find near optimal solutions to the bin packing problem: Next Fit (NF), First Fit (FF) and Best Fit (BF) (Corcoran and Wainwright, 1992; Graham, 1978). Considering the case of pipe cutting (Figure 14), the Next Fit strategy consists in taking a new pipe as soon as the object considered can not be cut in the current pipe. Previous pipes are not considered. The First Fit strategy consists in taking into account the free length of pipe available in all the previous pipes and cut the first from which the object can be cut. The Best Fit is a variant of the First Fit in the sense that all the previous pipes are considered and the one in which the loss-the remaining length after the current cut-is minimum, is cut.

Research involving these methods shows that the First Fit algorithm in conjunction with an ordering of the objects going from the largest to the smallest (First Fit Decreasing algorithm, FFD) gives, on average, the best answer to the 1D packing problem (Graham 1978). In 1973, Jeffrey D. Ullam proved that the FF algorithm can not perform worse than (17/10 opt + 2). 'opt' standing for the optimal number of pipes required to satisfy the orders

from customers. David S. Johnson of the Bell Laboratory proved that if, in addition, the objects are ordered in decreasing size, like in the FFD algorithm, then the near optimal solution found can not be worse than (11/9 opt +4) (Graham 1978).

In order to tackle the inherent complexity of such problems, some non-deterministic methods are beginning to emerge. Indeed, it is important to notice that the way common mathematical or engineering problems are solved does not include such factors as randomness. A problem is solved exactly or it is not. It can not be solved on average. This is precisely a new state of mind in today's methods, more suited for concrete cases, which accept that a method sometime solves the problem, and sometimes fails or accepts a near optimum solution in a reasonable time (Traub 1994). Corcoran and Wainwrigh (1992) programmed a Genetic Algorithm to solve a bin packing problem using up to 500 packages and thus broke all the limits ever considered for such a problem. They found that a Next Fit strategy performs better with their method.

These works constitute a typical aspect of the results recently obtained in computational mathematics. On one hand, the results provided by Ullam and Johnson remain true whatever the number of components is. However, the bounds proposed cannot be tolerated for industrial applications. On the other hand, the results of Corcoran and Wainwright are impressive by the size of the problem they addressed successfully, however, their method do not prove anything on the theory behind packing problems.

Squares into Square Containers

Packing squares is also very common. One of the most famous problem of this type, without any real application, consists in 'fitting a set of squares no two of which are alike into a larger square without any overlap or leftover space' (Gardner 1979). The only known solution to this problem is a square which side is 70 units in length because $70^2 = 1^2 + 2^2 + .. + 24^2$. Another well-known version of this problem consists in packing a large square container with smaller identical squares (Figure 16).

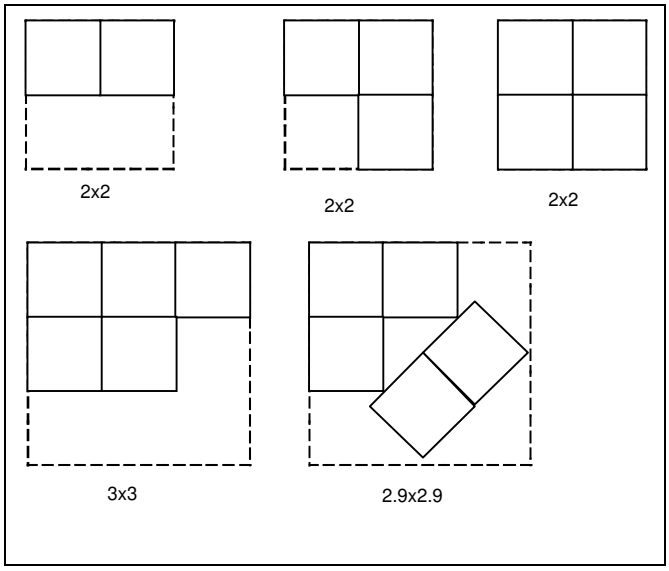

Figure 16. Packing squares into a square container. When the squares are rotated by 45 degrees the solution of the 5 cubes problem changes (Gardner 1979).

The exact solution is known for n = 2,3,5,7,19 and the squares of integer numbers. F. Godel in (Schrijer 1982) writes that "for small n" the upper bound and lower bound of the size of the enclosing square are respectively Ceil[Sqrt[n]] and Sqrt[n] where Ceil[x] gives the smallest integer greater than or equal to x. The main difference between these problems is the nature of the variables. In the first case variables can be discrete while in the second case they must be continuous.

Squares into Rectangular Containers

This case is of much more practical interest than the previous examples. It is encountered in the loading of trucks, planes, trains and boats. It is a particular case of the rectangular packing problem.

Rectangular Packing

This problem consists of maximizing the density of rectangles. The density is defined by the ratio of the sum of the rectangle areas with respect to the area of a rectangular container. In order to simplify the task, the sides of the rectangles are constrained to stay parallel or perpendicular to each other. This problem, also called constellation, has been studied for several types of configurations and upper bounds are known for the density of

some of these constellations. When a container is defined, typical practical applications are found in stock cutting (Figure 17), in its 2D version of the pipe cutting problem (Figure 14), plant/electronic layouts (Figure 19) and pallet loading (Figure 18).

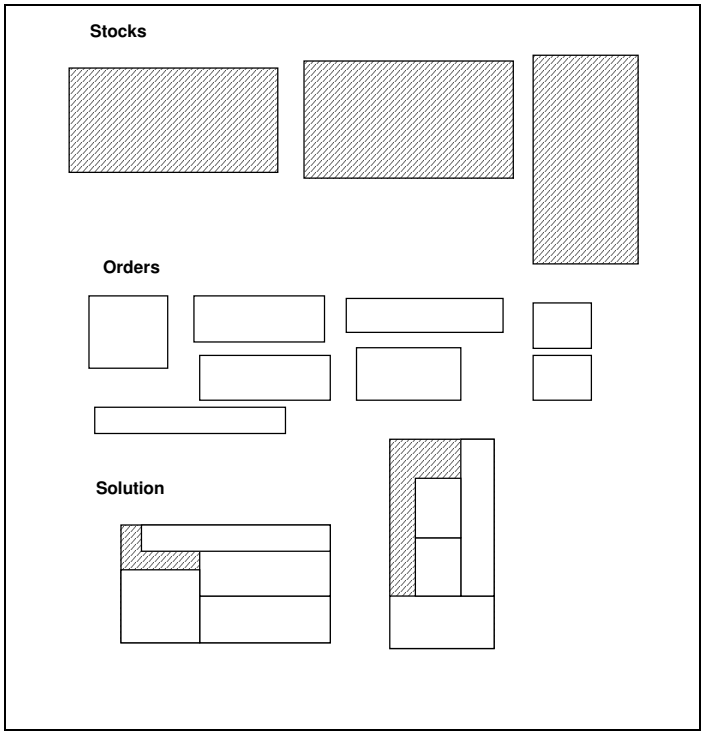

Figure 17. Rectangular stock cutting (Dyckhoff 1990).

The later example has been extensively studied. It consists in maximizing the number of pallets-which have the same rectangular shape and size- loaded into a rectangular container- i.e. a boat, a wagon or a truck (Dowsland 1993).

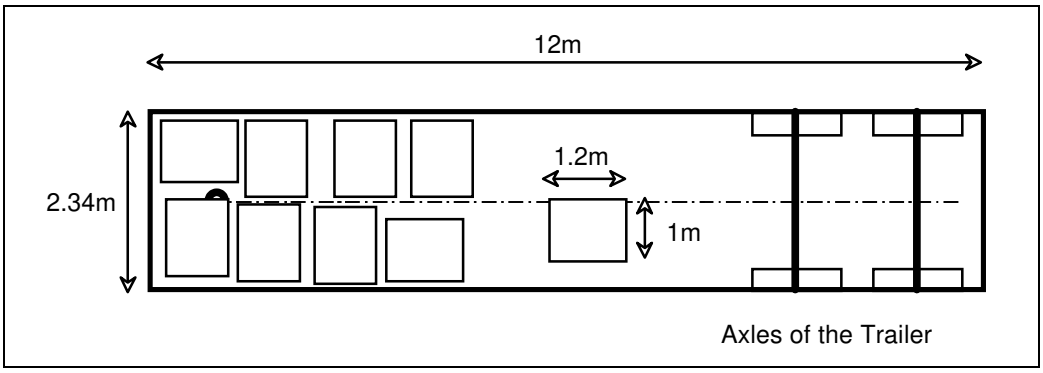

Figure 18. Trailer loaded with standard pallets.

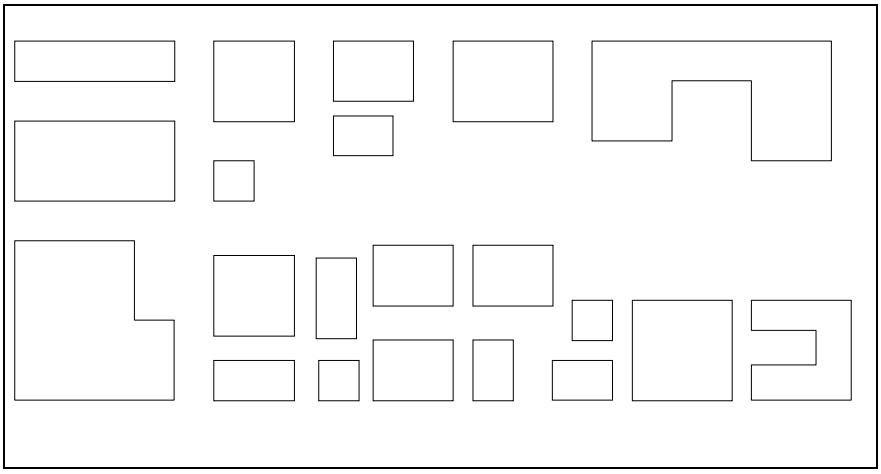

Figure 19. Plant Layout Design. This configuration problem involves several buildings to be placed in a limited area such that the path length between two related buildings is minimized (Stoyan 1983).

Many methods have been derived to achieve an acceptable level of compaction but none guarantees the optimum in finite time (Dowsland and Dowsland 1992). These methods involve different approaches such as dynamic programming, tree search or very specialized heuristics like finding optimal sub-configuration by assembling the objects in groups of two or three. As we will see in the next paragraph, global optima are generally not formed by a juxtaposition of local optimal configurations which makes the latter approach dedicated to special cases. Other attempts involve the use of Simulated Annealing and Genetic Algorithms. The reader can consult Dowsland (1992, 1993) for further references on these cases.

Circles, Cylinders and Spheres

The packing of these three types of entities is a particular case of minimizing an n-dimensional convex hull volume or surface around a set of n-dimensional spheres. When n=2, the spheres becomes circles, and their optimal convex hull is a geometrical figure as close as possible to an hexagon called Groemer packing only when the number of circles is in {0..120} U {$n=3*k^2 + 3*k +1$, k integer} (Figure 20). For other numbers of circles, very good approximations of the best configuration are known but no exact solution was ever found (Stewart 1993).

Packing cylinders can be reduced to the previous problem (Stewart 1993) whereas packing spheres can not.

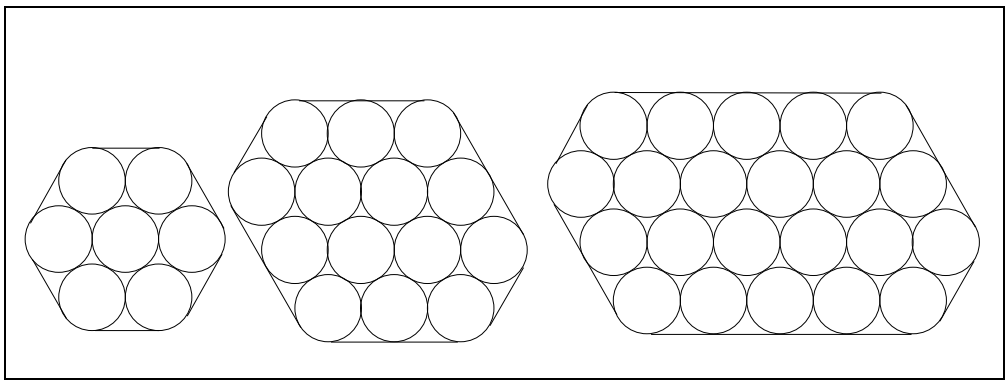

Figure 20. Groemer packing showing the configurations of smallest area (Stewart 1993).

If n=3 then the n-sphere is a common 3D sphere and the optimal convex hull is a sausage (Figure 21) provided that the number of spheres does not exceed 56. For a higher number of spheres, it is very difficult to find an optimal convex hull. It is conjectured that the optimum configuration is a compact shape as if all the spheres were put inside a larger globe (Stewart 1993).

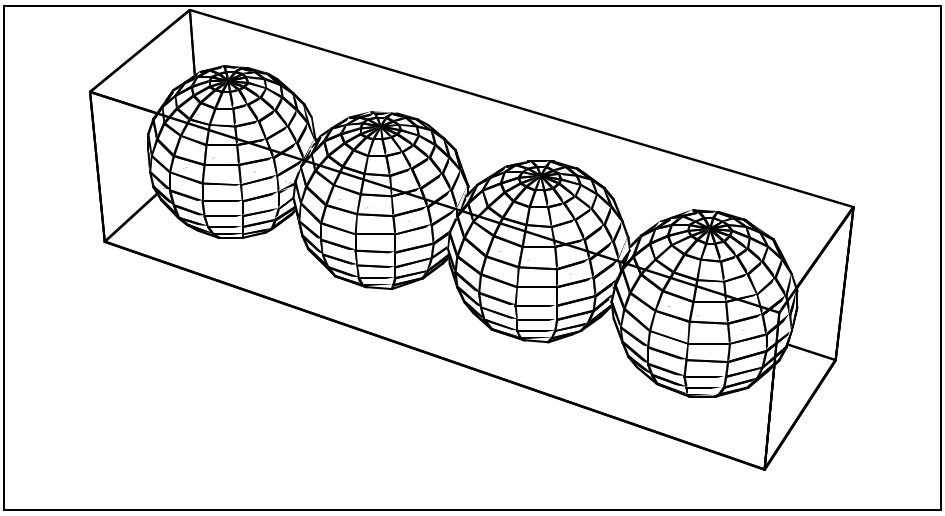

Figure 21. Optimum packing shape for minimizing the volume of the convex hull for less than 56 spheres.

This sudden change in the optimum configuration pattern when going from a 56 sphere CDP to a 57 sphere CDP is troublesome for algorithm designers. Indeed, the difference between the volume of a 57 sphere sausage configuration and the volume of the optimal arrangement might be very small and appear lately during the search thus providing no clue about which path leads to the best configuration.

A more difficult problem, but of more practical interest, occurs when the shape of the container is constrained. For example, what is the maximum number of balls one can fit into an a*b*c box? In all these cases the positions of the objects must clearly be continuous variables in order to be able to reach the optimum solution.

Free-form Packing

Free form packing refers to the packing of objects and containers having a non-regular shape. This case was not studied until recently because of two reasons. First, the complexity of the problem is such that even the best algorithms combined with the best computers could not give a satisfactory answer in a reasonable amount of time. Second, the methods of description of free form objects were not flexible enough to be handled conveniently by computer programs. Third, there was no theoretical interest in these cases. Kim and Gossard (1991), Szykman and Cagan (1994a, 1994b) and Lomangino (1994) used a

CAD software to represent the complex shapes and coupled such a program to an optimization software to find better configurations. Volume optimization was only one of their goals. They developed their own methods to constrain objects to assume given positions due to their functionality. This case is the most general of the Configuration Design Problems, hence its variables are continuous as well as its objectives.

Industrial Packing Problems

These problems are usually constrained, involve multiple criteria and a high number of variables. The shapes of the objects involved in these CDPs are unknown as well as the type of objective and constraints (linear, quadratic, non-linear, continuous or discrete).

Payloads Packing Problems

Payloads packing problems involve an additional objective as compared to mere volume optimization. Placing the center of inertia of the loaded container as close as possible to a given location (Aziz 1991; Amiouny et al. 1992; Wodziak 1994) (Figure 18) is of importance for planes and boats in order to reduce their fuel consumption and increase the safety of the trip. In addition to these benefits, a balanced loading reduces the damage caused by trucks to the roads.

Amiouny *et al* (1992) deal with a one-dimensional bin packing problem. The authors develop a heuristic to balance the load in an airplane. The heuristic packs the boxes in order of non-decreasing density to minimize the moment that is being generated at the target center of gravity. It does this by packing the least dense box, then calculating a new target point for the next box. The problem is then repeated with one less box until all boxes are placed.

Aziz (1991), presents a methodology for graphically representing boxes that are loaded into a truck using solid modeling techniques. A spatial occupancy enumeration technique is used to represent and test the boxes to see if they fit in the container. The technique allows for different sizes and orientations of boxes. These techniques are considered during the implementation of the graphical display of the boxes once the heuristic is completed.

In many of these cases, heuristics based on combinatorial optimization, are used to drive the optimization process. These heuristics have to consider some physical constraints

dictated by the problem, and thus must include constraints in the problem solution methodology.

Noticing that the truck packing problem is related to two classical operations research optimization problems, the knapsack problem and the Traveling Salesman Problem (TSP) Wodziak (1994) and Grignon and Fadel (1996) used a mixed combinatorial and continuous formulation to address this problem. The design variables being the distances between the boxes and the order in which the boxes are loaded into the truck, the non-interference constraint is removed from the problem only leaving the constraint of being inside the container. This last constraint was implemented using various strategies like interior, exterior penalty functions, and heuristics. These methods are dedicated to square like object shapes and thus are difficult to apply to engineering problems in which the objects are free form.

Accessibility, Maintainability, Disassembly

Only few recent works propose an automated computational solution to the accessibility problem. In the past years accessibility problems were addressed by using real models and costly simulations. More recently, Schroeder et al.(1995) use a CAD software to represent objects as well as the space required to move them. This space is called a 'swept surface' and is made of the envelope of the object when moved along a path. The object is removable if the swept surface does not intersect any other component of the system. The path is computed using an attraction field and a randomized method to avoid local traps. This technique is attractive but needs to be refined since it can provide very complicated paths including rotation of the objects along all the directions. An alternative to the approach of Schroeder et al. is to consider lines of equal potential in the design space (Hsuan 1996). In order to know if a part is removable, it has to be left loose to drift along these potential lines, attracted by a destination point outside the system. A collision check must be made in order to eliminate any unfeasible trajectory. This strategy has the advantage of being simple, however it requires the calculation of the potential lines around the current position of the object. For 3D freeform objects, this task might be difficult since the potential is the result of the interactions of all the components of the system. More over, this approach guarantees to find a path, but not necessarily the best one.

Research Opportunities in Configuration Design.

As described in the literature review, the research in Configuration Design was, until very recently, mainly concerned about Discrete Optimization and Geometric Combinatorics. Addressing the more general problem of Freeform Configuration Design requires the use of a different field of mathematics: non-linear continuous global optimization whose literature review is presented in the next paragraph. However, before going further, some interesting questions about configuration design alone must be emphasized. For instance, how can the complexity of a continuous configuration problem be characterized? For this purpose, some criteria of complexity have to be defined for this type of problems. Then, how can one use these criteria in order to help the optimization method to find a solution. A second category of questions concerns the information that an engineer might gather just by looking at the individual components and at the total system. Can he or she foresee the difficulties and make good choices in order to make an easily optimizable design?

Third, many questions concern algorithms applicable to freeform components. There is, in this new domain, a general lack of data on the efficiency of the existing methods. No framework of reference test cases exists.

Finally, a large part of the literature on packing problems concerns single objective optimization while Engineering Configuration Problems are known to be multi-criteria. Hence, additional research opportunity involves the formulation of the Configuration Multiobjective Configuration Problem and the search for the multiple solutions of the Pareto set.

Global Single Objective Optimization Literature Review

In this section, global single objective optimization methods are reviewed. They are classified into categories based on two characteristics: the use of randomness and the use of information gathered during the optimization process. This classification is summarized in table II.

The first characteristic, i.e. the use of randomness, can be described by the ability of the algorithm, if given twice the same input, to generate different results. In this case the method is said Non-Deterministic or Stochastic otherwise it is Deterministic. The second characteristic, i.e. the use of information, can be defined by the capacity of the method to use information previously gathered during the run and building strategies in order to progress

quicker. A 0-depth information use means that there is no information gathering. A pure Random Search is a 0-depth method. An N-depth information use means that a strategy is based on the current step and N previous steps in order to determine the next move. Total capitalization means that the totality of the information available is studied and converted into knowledge for the next move (this approach is closer to the integration process of the human brain).

Table II. Optimization methods classification. Among the local search algorithms, the type A groups all the methods comparable to steepest descent or crude dichotomy, the type B groups the methods comparable to conjugate gradients. It is to be noted that methods of type B capitalize efficiently on the information, but only for local search.

	Information Capitalization			
	0-depth	**2-depth**	**N-depth**	**Total Capitalization**
Deterministic		-Local search of type A	-Powell conjugate directions (N = number of variables)	-Local Search of type B -Surface Reconstruction
Non-Determinist	-Pure Random -Monte Carlo	-Multi Start + Local Search A -SA	-Multi Start + Local Search of type B	-GA -Rule-based systems

One of the more general formulations of a constrained optimization problem looks like the following (Huang-Min and Arora 1995):

optimize the objective $f(\mathbf{X})$; $\mathbf{X} = (x_1, x_2, \ldots x_n)$

subject to the following constraints

$g_i(\mathbf{X}) \geq 0$; $i = 1..p$ (inequality constraints)

$h_i(\mathbf{X}) = 0$; $i = 1..m$ (equality constraints)

x_i in D_i, $D_i = \{d_{i1}, d_{i2}, \ldots, d_{iq}\}$ $i = 1..n_d$ (discrete variables sets)

$xl_i \leq x_i \leq xu_i$; $i = n_d + 1 .. n_c + n_d$ (continuous variables range)

$n = n_c + n_d$

Industrial size problems-ranging from 10 to several thousands of variables, are investigated using numerical methods. Once a starting value is assigned to all the variables

(**X**), a computer can search for a better solution-provided that the initial point is not the optimum. The choice of a search algorithm depends on the type of objective and constraints-whether they are linear, quadratic, or else, on the number and types of variables (discrete or continuous), and on the topology and shape of the variable and objective spaces (disjoint, convex, non-convex).

It is beyond the scope of this work to enumerate all these methods. However, since Configuration Design Problems are clearly constrained non-linear, this part of the literature survey focuses on constrained non-linear optimization techniques and on unconstrained optimization techniques coupled with penalty functions.

Determinist Search

A large majority of the Deterministic search algorithms are based on the concept of the steepest descent and its improvement: starting at one point, they move toward a better point using the slope of the objective function. They figure in this literature review because they are part of more complex strategies mixing local search and non-deterministic methods. They can be classified according to their use of the objective function alone (0 order methods), first derivative (first order), and second derivative (Hessian matrix, second order) and to the existence of constraints. Indeed, it is intuitively understandable that a local minimum of an objective function can be found by 'sliding' along lines of steepest slope or by some variation of this strategy. Among the unconstrained first order methods, one can choose either conjugate gradients (Fletcher-Reeves algorithm, Beale's Restarted Conjugate Gradient Technique, Polak-Ribiere method) or quasi-Newton, also called variable metric methods (Davidson-Fletcher-Powell algorithm, Broyden-Fletcher-Goldfarb-Shanno (BFGS) method). One of the differences between these two categories is the memory requirement. Conjugate gradient methods require a storage of order N -the number of variables- and quasi-Newton methods require a storage of order N^2 (Press et al. 1990). When the gradient is not available, the values of the function itself must be compared at different points. A strategy is used to progress toward the optimum such as Powell's conjugate direction (Powell 1964) which is a 0-order method.

All the previous methods can be applied to solve constrained problems when their constraints are embedded as penalty functions into the objective. This approach is widely used in spite of the fact that it makes the objective function more complex. The penalty is

said to be an exterior penalty function if it is applied only when the constraint is violated. The main drawback of these functions is that they do not prevent the optimization process to converge toward unfeasible solutions since the difference between slightly unfeasible and feasible is small. On the contrary, interior penalty functions work only for feasible design points. They increase in magnitude when the design point gets closer to one of the boundary of the feasible domain, which makes the solution of the optimization always conservative. However this technique requires the identification of a feasible point (Haftka and Gurdal 1993). Extended interior penalty functions have the characteristics of both previous. A common technique, for problems involving a large number of constraints, consist in replacing the constraints by a unique expression like the Kreisselmer-Steinhauser function (Kreisselmer and Steinhauser 1979),

$$KS[\, G_i[X]\,] = -(1/r) * Ln\Big[\sum_{i=1\ldots m} Exp[-r * G_i[X]\,]\Big]. \qquad (1)$$

where r determines the closeness of the KS function to the smallest inequality

$G_{min} = Min\,[\, G_i[X]\,]$.

The values of the KS function are thus bounded by,

$$G_{min} < KS < G_{min} - Ln\,[\, m\,]\,/\,r. \qquad (2)$$

Other similar techniques involve augmented Lagrangian functions such as,

$$f[\, X, \lambda, \mu]\,] = f(X) + \sum_{i=1\ldots m} \lambda * G_i[X] + \Big[\sum_{i=1\ldots m} G_i[X]^2\Big]/(2*\mu).$$

Conn et al. (1994) present the application of such a method on numerous test cases described in the Constrained and Unconstrained Testing Environment (Bongartz et al. 1993). The main drawback of these techniques relies in the fact that the functions involved must be twice continuously differentiable.

Nevertheless, constrained optimization has its own methods among which Gradient Projection, Reduced Gradient (Rosen 1960) and Feasible Directions (Zoutendijk 1960) are the most common (Haftka and Gurdal 1993). More advanced techniques like Newton's methods make use of the Hessian Matrix in addition to the Gradient in order to progress quicker toward the optimum. As their penalty based counterparts, the main drawback of all these

algorithms is the use of the gradient which is usually expensive to calculate. Moreover, they can be applied to at least differentiable functions.

Non-Determinist Search

Most of these techniques are 0-order methods. One of the most famous of these methods is the Pure Random Search (PRS) because it allows drawing statistical conclusions about the problem at hand and can easily be coupled with Local Search.

Among the Probabilistic Search Algorithms (PSA), Genetic Algorithms, (Holland 1975; DeJong 1975) and Simulated Annealing (Kirkpatrick 1983) are gaining popularity. While SA simulates the atoms movements during the cooling of a metal, the GA relies on an imitation of the natural evolution of living beings which uses selection, reproduction and mutation processes to generate better individuals (solutions in Mathematical Optimization). The Genetic Algorithm for example, progresses from a set of 'parents' points to a set of 'children' points which in turn become parents. From generation to generation, the population migrates closer and closer to the global optimum of the objective function (Figure 22).

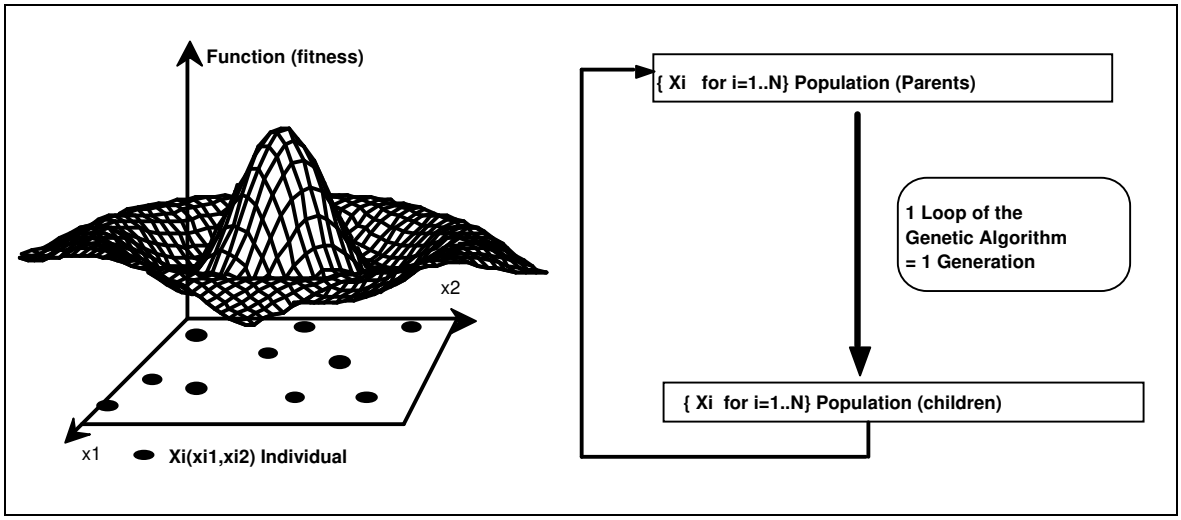

Figure 22. Genetic algorithm iterative process. From a population of parents the genetic algorithm evolves a population of children who in turn, become parents of the next generation.

The key to the success of this migration relies on a four step process applied to the members of the populations: coding, selection, reproduction and mutation (Figure 23).

The GA may be stopped based on two criteria, either when the maximum number of generation is reached, or when the improvement of the fitness is below a level imposed by the user during a given number of generations.

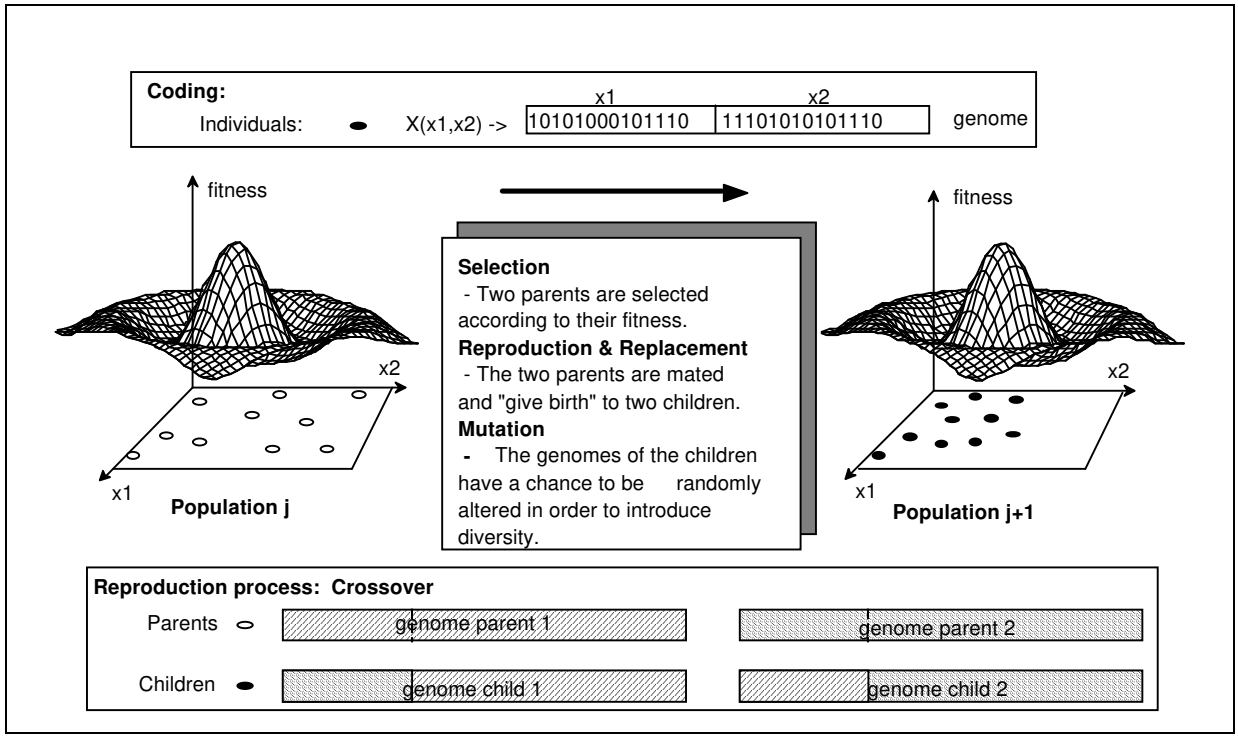

Figure 23. Details of Genetic Algorithms Mechanisms.

The first advantage of the evolutionary methods i.e. methods which are relying on the use of computational models simulating the mechanisms of living beings evolution (selection, reproduction, mutation), as compared to first and second order techniques, is due to the fact that the additional objective function evaluations typically required to compute the gradient vector or the Hessian matrix are not needed. Indeed, computing the gradient by finite difference requires the evaluation of N+1 objective functions, N being the dimension of the **X** vector. Computing a Hessian matrix is even more expensive and requires N*(N+1)/2 additional objective evaluations and brings the total computational complexity of the second order techniques to $O(N^3)$ (Haftka and Gurdal 1993, p. 138). This high computational cost fostered the development of Quasi-Newton methods which make use of the Gradient to build

an approximation of the Hessian matrix. Finally, the total number of objective evaluations increases linearly in function of the number of variables for first order methods and quadratically for the second order methods, which can be overwhelming for problems with high number of variables or computationally expensive objective and constraints functions. The PSA have the second advantage of being less prone to fall into local minima wells present in multimodal functions. DeJong (1975) showed the superiority of GAs by using a set of five non-linear, multimodal and sometime non-differentiable or non-continuous functions. PSA can also handle disjoints domains and integers variables. Beaslay *et al*. (1990), and Mahfoud (1994, 1995) extended the possibilities of Genetic Algorithms in order to handle functions with several global optima.

Even among the PSA category differences appear. For example, GAs are able to handle mixed discrete/continuous as well as fully discrete or fully continuous variables without any change in their definition while SAs present more differences for handling each case (Eglese 1990, Dekkers and Aarts 1991).

The drawbacks of these techniques are the tuning up of their parameters in order to converge quickly and the estimation of the accuracy of the result (no optimal solution, even local, is guaranteed) since their stopping criteria do not involve slope or curvature information but only number of function evaluation or steadiness of the best solution reached so far. Moreover, for fairly regular functions, as well as a low number of variables, they are computationally more expensive than deterministic methods.

Hybrid Determinist-Probabilistic Search

From the methods presented above, which are among the pillars of numerical non-linear optimization, emerged some hybrid techniques which couple the power of both PSA and determinist algorithms.

The multistart method uses either a PRS or a PSA to generate starting points for zero, first or second order local search and thus improves considerably the quality of the optimum found by the either the PSA or the PRS alone.

Clustering methods were developed to prevent multiple Local Search from converging toward the same local optimum (Becker and Lago 1970). Many works extended this idea exploring various sampling strategies, different clustering techniques and several Local Search. In 1990 Clustering methods were enriched by the Topographical Global Optimization

(Julesius 1989, Viitanen 1991, Torn and Viitanen 1992). However, it seems that these algorithms remain efficient for a small and medium number of local optima.

Hart (1994) compared the performance of a Local Search coupled first to a PRS, then to a GA (belonging to the PSA category). The GA using systematically a Local Search for each child performed better, however the number of objective functions evaluation was very high. In order to reduce this number he also made the hypothesis that it was not necessary to run a Local Search for each point generated by either the PRS or the GA. He concluded that this was not always true. On one hand, if the local optima of the function reflected some domain-wide characteristics (Figure 24) it was advantageous to restrict the number of Local Searches.

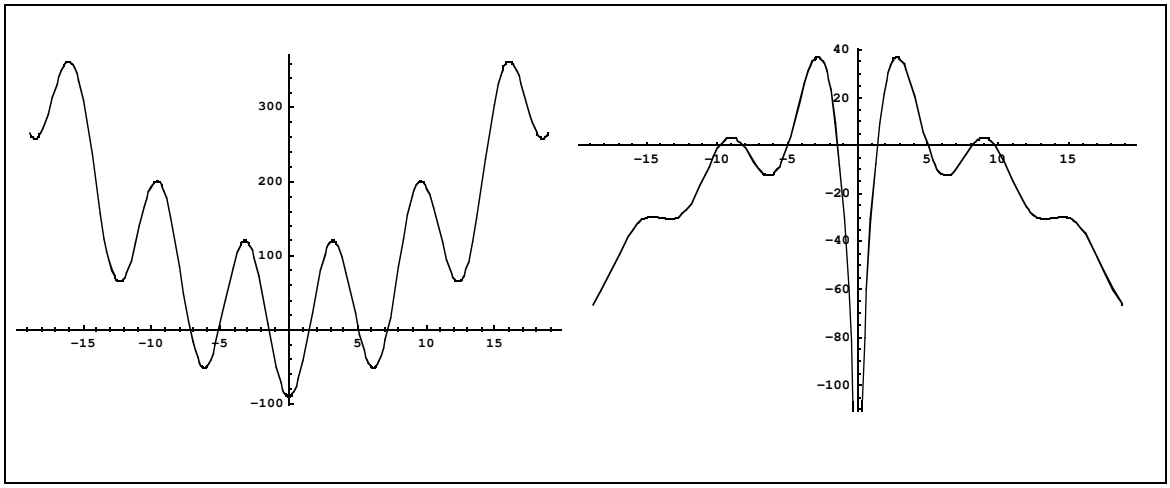

Figure 24. Domain wide characteristics reflected by the Local Optima. At left, the local optima decrease in the direction of the global minimum, at right the inverse phenomenon happens which is armful for a method trying to learn from local optima.

On the other hand, when this is not the case, it is more advantageous to run a Local Search for each point. He was unable to clarify the notion of 'domain wide characteristics' for multi dimensions functions and thus the relation between the topography of the landscape and the LS frequency remains unclear. A second conclusion of his work is that using Elitism, for the GA evolution, improved greatly the results. Moscato (1992), in order to implement his idea of Memetic Algorithm (MA) based on the concept of 'memes' (small coherent units that can be assembled in order to build a larger system), uses a GA coupled with a SA to do

the Local Search. Like in (Hart 1994), he found that the success of the MA is mainly due to a correlation of position of the local optima with the position of the global optimum of the function. Still the effect of this correlation on the optimization results remains unexplored, thus no conclusion can be made about the adaptability of such methods if more information about the local minima is available.

Research Opportunities in Optimization and Configuration Design

Configuration problems are numerous and heuristics to solve them are at least as numerous, each taking advantage of specific characteristics of the CDP to solve it more efficiently. However, engineering design removes, from the CDP formulation, the previous knowledge of the system characteristics (like components shapes) so useful for making simplifying assumptions such as in the standard squares and circle packing problems. This constraint, in addition to more theoretical reasons, fosters the use of non-deterministic algorithms sometimes combined with conventional optimization techniques.

In practice, the engineer changes the parameters of the product, i.e. the shape of the parts and their configuration, to optimize some objective function like cost, weight, robustness, sensitivity to vibrations, deflection, heat generation, safety, ergonomics, or aesthetics. The engineer's experience and capacity of reasoning restrict the possible solutions to few choices. This is not accessible to common automated optimization methods that do not take advantage of any type of past knowledge about the system's behavior and manufacturing process. Hence, a first area of research involves the introduction of knowledge in the optimization process in order to guide the search. In this area, the collaboration between the engineer and the method must be studied. Black box optimizers are not well perceived in the engineering fields.

Non-deterministic methods have been extensively used on problems involving a restricted number of well-specified configurations (component selection problem) (Dixon 1986, Orelup 1987, Ramachandran 1988) however their application to large scale engineering configuration problems involving continuous variables still belongs to the academic field. Hence, a second interesting area of research involves the type of information that a search algorithm would require in order to be applicable to non-discrete configuration problems and the efficiency of such a method.

The best-known packing problems are those involving volume, surface or length. One can wonder if the methods used to solve such problem are still applicable to solve a CDP with a different objective. Is there equivalence between problems involving different objective functions? What are the factors that can be used to compare the CDPs as optimization problems?

Mechanical assemblies are made of hundreds to thousands of linear and non-linear geometric constraints that translate construction and functional imperatives. It is doubtful that the constraint satisfaction problem solved to create these assemblies could be augmented (and complexified) with multi objective optimization. Hence, performing Configuration Design Optimization means that a new way to handle constraints must be defined.

From the point of view of the data type, deterministic methods are restricted to ordered variables and connected domains while non-deterministic methods can handle disconnected domains and non-ordered variables, providing them a definite advantage for dealing with discrete configuration choices (Dixon 1986, Orelup 1987, Ramachandran 1988). However, CDPs are not fully discrete which does not exclude the use of deterministic methods. This cooperation is another open area of research.

Multiobjective Optimization Literature Review

Given a function $\mathbf{F}(\mathbf{x})$ where $\mathbf{x} = (x_1, x_2, .. , x_n)$ is a vector of a domain D of R^n and $\mathbf{F}(\mathbf{x})$ is also a vector $\mathbf{F}(\mathbf{x})=(f_1(\mathbf{x}), f_2(\mathbf{x}), .. , f_m(\mathbf{x}))$ the Multi Objective Optimization Problem (MOP) can be stated as follows:

F : **x** -> **F**[**x**]
$\mathbf{R^n}$ -> $\mathbf{R^m}$

Find one **x** such that **x** is non-inferior.

A solution $\mathbf{x_1}$ is said to be a non-inferior solution of the MOP if there exists no other vector $\mathbf{x_2}$ such that $f_i[\mathbf{x_1}] \triangleright f_i[\mathbf{x_2}]$ for all i with a strict inequality for at least one value of i. The solution of such a problem is usually a set of points.

Conventional Multiobjective Algorithms

Multiobjective optimization really took off after World War Two, especially after the publication, in 1951 of the necessary and sufficient condition for non-inferiority by Kuhn and Tucker (Kuhn and Tucker 1951). Since then, the mathematical methods to find a solution to an unconstrained multiobjective problem can be put into one of the following categories (Statkinov and Matusov 1995):

- Substitution of all the criteria by a single one (scalarization) either by choosing the most important, or by building a combination of all the criteria into a single one. In this last category one can find the weighting method (Zadeh 1963) in which all the criteria are replaced by their weighted sum, the kth-objective Lagrangian problem and the kth-objective ε-constraint problem (Chankong and Haimes 1993).
- Consecutive optimization of all the criteria.

In the first category, the weighted sum method finds a single solution in the Pareto set for each set of weights. The restrictions on the differentiability and continuity of the objective and constraint functions depend on the optimization technique chosen to solve the new problem. Independently of this first restriction, the weighted sum method is unable to find Pareto points inside a duality gap (Figure 25). The other classical methods enumerated present the same drawbacks. A description of these methods can be found in Chankong and Haimes (1993).

Several other approaches to solve multiobjective optimization problem can be found in the literature among which Goal Programming first studied in depth in the work of Levary (1984), and Compromise Programming.

Recent efforts were made to find methods applicable to non-convex Pareto sets. Kostreva and Wiecek (1993) proposed a method for non-convex Pareto sets generated by linear complementary problems. TenHuisen and Wiecek (1996) use a non-linear scalarization based on a quadratic Lagrangian function for producing locally efficient solutions.

The major drawback of many of these techniques comes from the strong assumptions made on the continuity and differentiability of the objective and constraint functions especially when Kuhn and Tucker conditions are used in the process. These assumptions are difficult to guarantee in Engineering Configuration Design. A second drawback comes from the fact that they offer a single solution to the Multi-objective optimization problem.

Recent Multiobjective Global Search Algorithms

Recent trials to obtain multiple solutions to MOPs in the field of engineering involve the use of parametric curves that approximate the Pareto set (Rokowska et al. 1991, Rao and Papalambros 1989). These techniques are restricted to bi-objective problems. Das and Dennis (1996) propose a Normal Boundary Intersection method (NBI) in which the original problem is replaced by a sequence of alternate formulations solved by a weighted method. This strategy provides very good results as long as the boundary of the Pareto set is not folded i.e. does not contain any duality gap (Figure 25).

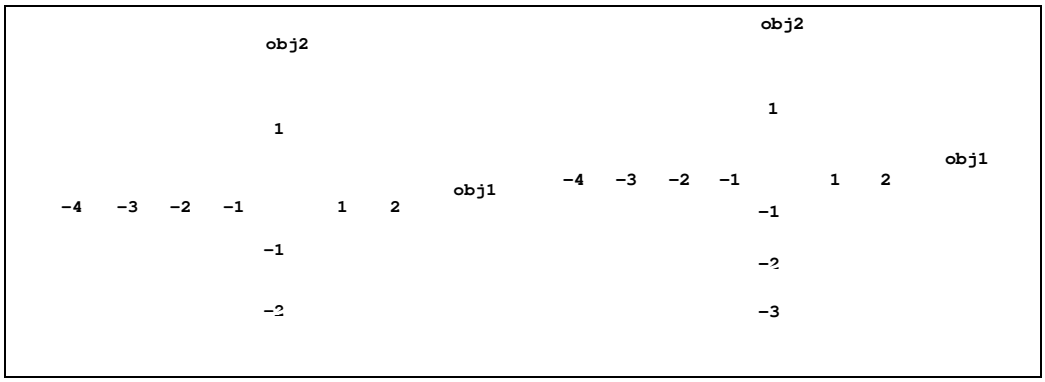

Figure 25. Examples of duality gaps with different depths.

The coupled use of genetic algorithms and multicriteria optimization is still very rare (Schaffer and Grefenstette 1985; Venugopal and Narendra 1992; Grignon and Fadel 1996, 1997).

The major problem encountered by researchers trying to use GAs to obtain multiple solutions at the same time is the tendency of GAs to bring points of their population together. Thus many solutions were proposed to counter this effect. For example, Schaffer (1985) proposes to select sub-populations of points proportionally to each objective in turn. For M objectives, M sub-populations are generated with an equal number of individuals. The final population is obtained by shuffling all the sub-populations. Unfortunately, this algorithm tends to converge to few extreme members of the population after a large number of iterations. Schaffer then tries to prevent this convergence from happening by mating far-away

individuals. This mechanism is not selective enough to prevent poorly fitted individuals from mating with good individuals. Finally, he concludes that random mating is more efficient.

Baker (1985) uses a single criterion based on a ranking of the individuals. The points are first sampled and all non-dominated points are removed from the population. Goldberg (1989) points out that 'this should be used in conjunction with niche and speciation for stabilizing the multiple sub-populations arising along the Pareto-optimal front and thereby preventing excessive competition among distant population members. Richardson et al. (1989) argue that this is equivalent to linearly combining the objectives similarly to the weighting method with different weights at each generation. As for Schaffer's method, this leads to speciation.

In spite of this drawback, Horn and Nafpliotis (1993) used a niched Pareto GA through a combination of two methods: Multiattribute Utility Analysis (MAUA) and GA to deal with multicriteria optimization problems. Two approaches were chosen: either using MAUA in order to combine the criteria and then optimize with GA or use GA to find the Pareto set, and eventually use MAUA to select among the Pareto points. In order to find the Pareto set, they relied on the Niched Pareto Genetic Algorithm which uses a new kind of tournament selection: Pareto domination tournaments with sharing, since Oei et al. (1991) showed that niched GA with conventional tournament selection does not give steady results.

In order to avoid the previous drawbacks, Fonseca et al. (1993) propose a selection scheme based on the definition of the Pareto set which ends up with a ranking of the points based on how many points dominate the current point. However, this later method does not differentiate individuals that are close to the Pareto set from those that are further away. They also use sharing in the objective space instead of in the variable space in order to get a uniformly distributed population in the objective space. They calculate the sharing parameter σ_{share} using an over estimation of the size of the Pareto set in the objective space. Then, they note that mating restrictions σ_{mating} (a parameter similar to σ_{share}) which help to avoid excessive competition between the distant members of the population may not be efficient due to the respective low number of points in the population with respect to the area of the Pareto set. In order to prevent this drawback, they use higher-level decisions (a decision-maker) in order to focus on some areas of the Pareto set. Finally, Osyczka and Kundu (1996)

present a method based on a fitness calculated using a distance of the current point in the objective space with respect to a Pareto set updated during the run of the GA.

Two points from this literature review are interesting. First, a large part of the efforts spent in the domain of Genetic Algorithms and MOPs were conducted in order to address two problems:

- Find a fitness function that translates the multiple criteria into a single one.
- Prevent the GA population from collapsing to one or few points.

Next, in most of the examples encountered, the Pareto set is so huge with respect to the total space size that a random search with bookkeeping of the Pareto solutions would also give good results. It seems that most of the search effort is spent to evenly distribute the points in the final set rather than to find the Pareto set.

Sobol (1992) gives a final method, especially suited for Computer Aided Design. The method consists of 3 steps and only uses optimization for refining the final decision of the engineer. The first step consists of computing the values of the objectives at some selected points belonging to an Lpτ sequence because these points constitute a much better sampling of an hypercube than the corners of the rectangular lattice (for a definition of these sequences of points see (Sobol 1967, 1990)). The computed values corresponding to feasible points are put into tables from the best to the worst. During step 2 a decision-maker (the engineer) sets some acceptable bounds for the objectives. Step 3 consists in checking if some points of the tables satisfy the bounds on the objectives. If the answer is positive then the search is stopped. Otherwise the engineer is asked to relax the constraint values of the objectives.

The main difference of this method with respect to the others is the translation of the optimization problem into a constraint satisfaction one. The objectives are not optimized but reach at least a value driven interactively by the user. This is a 0-order method relying on an efficient space sampling (Lpτ sequences) and incorporating the knowledge of the user in order to drive the search. This seems to be a method well adapted to the engineering fields in which the magic black box, which gives the solution to the problem just by pushing the button, is not well accepted. However, when the number of variables increases this method might be expensive.

Research Opportunities in Multiobjective and Global Optimization

The literature shows a clear preference for the single solution approaches where multiple solutions are expected. Two approaches can be used to obtain these multiple solutions: either apply several times a method providing a single solution or apply a method providing several solutions at the same time. The difference between these two methods remains to be studied. Two main areas of interest for engineering applications can be identified. First, the automated domain search and second the cooperation of the engineer with the method. Concerning the space search, the author thinks that a good algorithm should be able first, to find the Pareto set and second, to cover as much as possible of this set. It should also be interesting to consider the use of Lpτ sequences of points whenever a local search must be made. It was also noted that it is difficult to identify the limits of the methods presented in the various papers. The authors usually claim that their algorithms behave well when the number of variables increases but it was already shown (Grignon and Fadel 1997) that the number of variables has much less effect on the performance of a searching algorithm than the size of the solution space with respect to the total space size to be investigated. Hence, building a method which is not, or is less sensitive to the size of the goal is a challenge. Finally, the interface engineer/search has its importance for engineering applications and the different ways to offer this collaboration is another open field of investigation that very few authors address.

Summary and Position of the Problem

Globally, Configuration Design Problems are non-linear in different ways. First, when moving the system components around, the objective functions might exhibit non-linearities. Second, when scaling the problem, i.e. when going from N components to N+1 components, the characteristics of the solution can drastically change. Thus, it is dangerous to learn from smaller size problems and to try to generalize the findings to bigger size cases. For this last reason, it is then recommended to handle all the components at the same time. This is the main criticism to all the heuristic based approaches applied to solve conventional packing problems. Hence, most of the global algorithms cannot guarantee a level of quality of their results. In the case of the CDPs, even the Approximation Algorithms (Schulz 1997) that are known for providing worst and average cases bounds for optimization problems, would be difficult to design due to the unknown shapes of the objects. A second criticism,

applicable to a majority of the methods reviewed, concerns the interaction with the user. The algorithms are magical black boxes providing the solution when the user turns the crank. The means for incorporating knowledge in the solving process is either non-existent, relies on the choice of the variables (case of the GA), or is incorporated in the heuristic (like in the heuristic used to solve scheduling or rectangular packing problems). There is no possibility to add knowledge during the run. This type of solutions in which the user is not in control is to be avoided for engineering applications.

It seems that non-deterministic methods are taking more importance than rule-based heuristics because they are more general and provide better results on average. While the first of these advantages can be justified by the definition of each method, the second is tied to the type of problem. For CDPs in general it is not clear whether the difference in the quality of the results is linked to the objective function, to the objects, to the number of variables, or to all these factors combined.

Identification of the Needs

The survey of the literature showed so many areas of interest and missing pieces of knowledge that it would take several books to cover all the topics. This work will focus on the few following needs related to Multi Criteria Optimal Configuration Design.

First, a formal general formulation of the configuration design problem could help describe precisely, analyze and synthesize the knowledge about the CDPs and eventually provide a common representation for all the configuration problems.

Second, a complexity analysis based on the former formulation of configuration problems would be welcome and help to build a classification in order to compare different CDPs.

Third, a general method for which the heuristic does not have to be redesigned for each objective or each type of components is missing. This method should be able to handle all types of objects ranging from simple cubes to freeform components.

Finally, all the CDPs presented in the literature review are single objective. However ECDPs are multicriteria. Hence, a method able to provide a set of points, dispatched along the Pareto set, would be welcome in a case as general as the ECDPs.

CHAPTER III

A GENERAL CONFIGURATION DESIGN OPTIMIZATION METHOD

Introduction

The CDOM intends to be able to deal with multi objective freeform Configuration Design Problems (CDPs). It was shown in the literature review that Multi Objective Optimization Problems (MOPs) are usually dealt with as single objective using a scalarization technique. Hence, it is proposed, in this chapter to adopt this same SOP formulation to be able to cope with freeform Engineering CDPs (ECDPs). Consequently, a single objective is taken into account and constraints are translated into penalty functions. All the ECDPs specific constraints are reviewed. Then, based on this single objective formulation, four measures to rate the CDPs complexity with respect to each other are defined. Several common CDPs are reconsidered and the behavior of several basic search algorithms is also studied in function of the evolution of these measures. This study is used to identify the influence of the formulation on the complexity of a CDP. Finally, several hypotheses are formulated on the definition of the CDOM and on the existence of several strategies which can contribute to the amelioration of the method.

General Configuration Design Problem Formulation and Complexity Measures

The literature review showed that the possible placement of some CDPs components requires non-integer positions (Chapter II, Squares, Spheres). These CDPs can no longer be addressed as combinatorial geometry problems. This fact is reinforced when the CDPs components become freeform. Placement variables must be real and no longer integer. One of the conclusions drawn from the literature review is that translating a continuous CDP into a discrete one is impossible in the general case. Hence, Continuous CDPs cannot benefit from the heuristics developed for their discrete counterparts. However, all the CDPs can be translated into a set of continuous functions of real variables that are the components of a constrained optimization problem (COP). This COP can then be translated into an unconstrained optimization problem in several ways among which the most common is the penalty function approach (Haftka and Gurdal, 1992). Although choosing this formulation results in a combination problem/method that does not bring results as efficiently as if rule

based specially designed heuristics were used, it has the advantage of being applicable to all the CDPs whatever the type of constraint is. Then, finding the best configuration for the system is equivalent to finding the global optimum of a new function made of a combination of the objective and of the penalties. Solving a multi objective optimization problem often requires the use of a scalarization which reduces the number of objective functions to one. Hence, the assumption is made that both constrained optimization problems and multi objective optimization problem can be reduced to a single objective optimization formulation. In this context, two types of information are of interest to compare the various problems and to make predictions about the efficiency of the solving process: information about the variable space (like its size), and information about the objective and variable spaces combined (bounds of the size of the solution, bounds on the size of the feasible domain, non-linearity). In the following, it is proposed to quantify this information with four criteria: roughness, trend, dilution of the feasible area, dilution of the solution.

Global Mathematical Optimization Formulation

Given :

- A global Cartesian coordinate system.
- A set of N objects defined by their shape (S) and positioned in space by 6 variables i.e. the 3 Cartesian coordinates of a particular point (x, y, z) and the 3 angles defining their orientation (α,β,χ).
- A set of equalities, usually called functional constraints H_k, positioning the objects with respect to each other.
- A set of inequalities (G_j)translating the notion of interior and exterior of the objects.
- One objective function of the 6N variables to optimize , F.

Find :

A set of values for **x** (a vector composed of all the positions of all the objects of the system) optimizing F.

Satisfying :

The equality and inequality constraints between the variables.

Notations

The objective function F, the inequality constraint G and the equality constraints H being functions of mixed integer and real variables regrouped into a vector **x**, the Constrained Optimization Problem can be formulated as follows:

$F : \mathbf{x} \rightarrow F[\mathbf{x}]$
$\mathbf{x} = \{ x_i \}\ i = 1 \ldots n \quad \text{with } n = n_c + n_d$

$G_j : \mathbf{x} \rightarrow G_j[\mathbf{x}]$
$H_k : \mathbf{x} \rightarrow H_k[\mathbf{x}]$

$\mathbf{x} = \{ x_i \}\ i = 1 \ldots n \quad \text{with } n = n_c + n_d$

n_c is the number of continuous variables
n_d is the number of discrete variables

$n <= 6\ N$

where N is the number of objects.

$xl_i < x_i < xu_i$ for $1 <= i <= n_c$

$x_i \text{ in } D_i = \{ d_{i1}, d_{i2}, \ldots, d_{im} \}$ for $n_c+1 \subset i \subset n_c + n_d$

Find one **x** such that

$F[\mathbf{x}]$is optimum and satisfies

$G_j[\mathbf{x}] < 0$; for all 'j'and $H_k[\mathbf{x}] = 0$; for all 'k'

This problem can be formulated as an unconstrained global optimization problem in which the constraints are combined with the objective function in order to build a single objective function. The integer variables can be extended to the real variable space i.e. an angle which can take few discrete values can be translated into a real variable whose range goes from 0 to 360 degrees. The formulation of the new optimization problem becomes:

$FP : \mathbf{x} \rightarrow FP[\mathbf{x}] + C[\ G_j[\mathbf{x}]\ ,\ H_k[\mathbf{x}]\ ]$
for all j and all k.

G_j : **x** -> G_j[**x**]
H_k : **x** -> H_k[**x**]
C : penalty function.

x = { x_i } i = 1 .. n with $n = n_c + n_d$
n_c is the number of continuous variables

$n <= 6N$

where N is the number of objects.

$xl_i < x_i < xu_i$ for $1 <= i <= n$

Find one **x** such that
FP[**x**] is optimum.

The following operators and quantities can then be defined.

Global Search: algorithm which associates to a function F its Global Optimum over its Definition domain [B_F].

GS[F] : F -> F-Opt = Opt[F[**x**]], **x** in [B_F].

Note: to the value F-Opt corresponds a set of **x**. This set could be either discrete, noted { **x**-Opt_i } (even reduced to a singleton **x**-Opt) or continuous, and noted [**x**-Opt]. In the case of a set of continuous sets it will be noted { [x-Opt]$_i$ }.

Local Search: algorithm which associates to a function f, the set of its Local Optima over its Definition domain [D_F].

LS[F] F -> { F-$Lopt_i$ } = { F[**x**] | F[**x**] is a local optimum and **x** in [D_F]}.

Note: to one value F-$Lopt_i$ corresponds a set of **x**. This set could be either discrete (even reduced to a singleton) or continuous and will be noted either { **x**-LOpt } or [**x**-LOpt] (Figure 26).

property : F-Opt belongs to { F-$Lopt_i$ }.

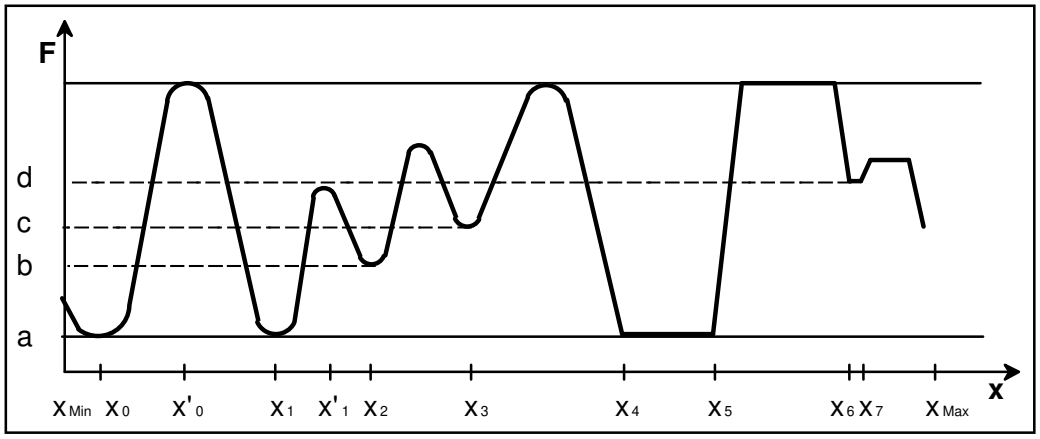

Figure 26. Example of multiple types of optima in the variable space and in the objective space. $[B_F] = [x_{min} , x_{max}]$, $[D_F] = [B_F]$, **GS** [F] = a, **LS** [F] = {a, b, c, d }. {**x**-opt} = $\{ x_0, x_1 \}$, [**x**-opt] = $[x_4, x_5]$.

Variables Hyper Box : the n-dimensional box defined by the Cartesian Product of interval of variations of all the variables of a function F.

notation : $[B_F] = \prod_{i=1..nc} [xl_i, xu_i] \prod_{j=nc+1 .. nd} D_j$

Feasible Domain : a subset of $[B_F]$ corresponding to all the variable values satisfying the constraints G_j and H_k of an optimization problem.

notation : $[D_F]$.

Unfeasible Domain : a subset of $[B_F]$ corresponding to all the variable values that do not satisfy at least one of the constraints of an optimization problem.

notation : $[U_F]$.

property : $[B_F] = [D_F] \cup [U_F]$.

Cardinal of a discrete set: the number of elements in this set.

notation : **Card**[{ }].

Discretization Function :

notation : **Disc**[B_F , m] : $[B_F]$ |-> $\{ B_F \} = \{x_{11}, x_{12}, \ldots x_{1m}\} \underline{*} \{ x_{21}, x_{22}, \ldots x_{2m}\} \ldots \underline{*} \{ x_{n1}, x_{n2}, \ldots , x_{nm}\}$.

Function Continuity type: C^0, C^1, C^2.

Cartesian product: $\prod$ or *

Based on this formulation and on the previous definitions four characteristics are proposed that allow the comparison of the CDPs.

Landscape Complexity Analysis

Translating a constrained optimization problem into an unconstrained one using penalty function is equivalent to transforming the variable space from a domain with forbidden areas into a domain where there is no forbidden areas. However, the penalty function transforms the morphology of the objective function graph by introducing additional non-linearity where before there were only the upheavals of the objective alone (Figure 27).

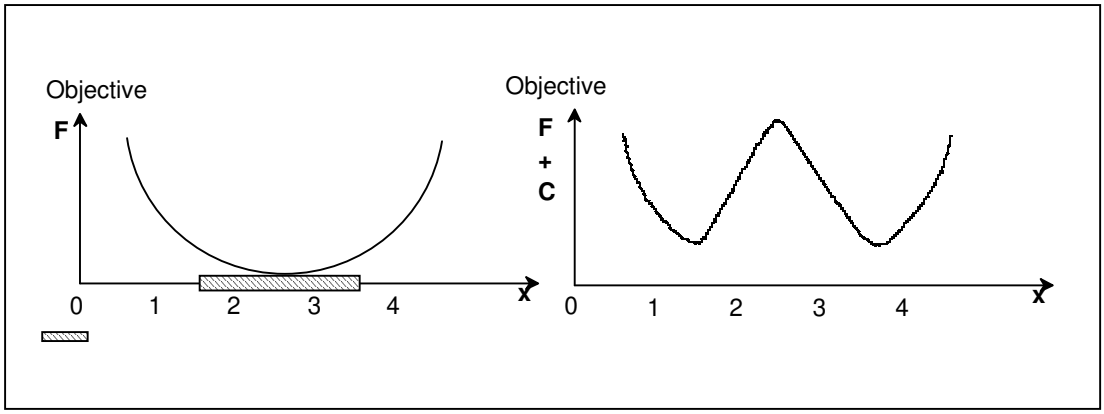

Figure 27. Constrained optimization problem landscape. The unfeasible areas is seen as forbidden areas (at left) and as penalty influencing the objective value (at right).

While the complexity of discrete CDPs is only a function of the number of their components, the complexity of continuous CDPs requires more factors to be evaluated.

Ko (1991) provides a complete study of the complexity of real functions in his book "Complexity Theory of Real Functions" in which he decomposes complexity into two factors: numbers computation complexity and functions computation complexity. In the current work

the assumption is made that the numbers computation complexity can be neglected since all the variables have a finite low accuracy. Only remains the functions computational cost among which the Max function must be taken into account. A second approach, called Information based complexity, developed by Packel and Traub (1987) is strongly related to the work of Ko. Both methods can provide bounds on the accuracy of an optimization problem provided that some regularity assumptions can be made on the derivative of the objective functions. This assumption allows first to know with which step the variables must be sampled in order to have the certainty to be at a distance ε from the global optimum and second, given the computational resources, to predict how accurate a solution will be. Unfortunately, even if the assumptions can be satisfied for the continuous objective functions such as the distance from the center of gravity and the inertia of a system, it is impossible to obtain such a guarantee for the penalty function. This makes their approach impracticable for ECDPs.

These considerations drive the search of complexity measures back to discrete complexity. Based on the notion of landscape, it is proposed to measure the difficulty of finding the value of the global optimum (F-Opt) of a function using two factors, one associated with the goal itself (the optimum of the function) and the second associated with the environment that is the function **morphology**.

As a metaphor for these two factors one can think about looking for a needle in a haystack. The bigger the needle, the more chance it has to be discovered. The second factor can be associated to the haystack itself (the environment). If all the hay twigs are arranged such that they all point toward the needle, it is easy to find the latter whatever its size, however if the latter twigs are randomly arranged then the size of the needle becomes an important factor.

In the following section, these quantities are defined using four criteria, and studied on a simple test case. Being tied to the function morphology, these factors can be applied to all the CDPs without distinction. The only assumption made is that $[B_F]$ is discretized such that the features of the function i.e. its local optima are always present in the sampled version of the problem.

Definitions

The function **landscape** and its local minima and maxima can be defined first by using an analogy with a landscape of hills and valleys. The landscape is the graph of the function submitted to the search method. A **valley** is the set of all the points of the landscape from which a rolling ball will reach the same set of connected low points (called floor of the valley). In the example of Figure 26, the number of valleys is 7 and the number of hills is 7. A ball with a starting position of x_0‘ + ε (ε positive very small) will fall in the direction of positive x and will eventually stop at x_1. Hence, all the points between x'_0 and x_1 belong to the same valley. Similarly, a ball starting its run at x'_1 - ε will fall in the negative x direction and eventually stop at x1. All the points between x_1 and x'_1 are in the same valley. Hence, the interval $]x'_0, x'_1[$ is a local valley whose floor level is $F[x_1]$. In this landscape, a **plateau** is a flat region where the function reaches a local optimum like inside the interval $]x_5, x_6[$.

It is now possible to define the three criteria used in this work to rate the complexity of a landscape i.e. the Dilution of the global optima, the Roughness of the landscape and the Trend.

Dilution of the Optima

The first criterion chosen to rate the difficulty of a CDP is the size of the solution (the size of the needle) in the variable space with respect to the size of the total variable space. This is called the Dilution of the solution (**Dil_S**).

Dil_S = Number of solutions / Total space size. (3)

It is equivalent to the probability of finding the solution by randomly picking a point in the variable space. To evaluate this criterion, each variable space must first be discretized. Then the expression of the dilution becomes

Dil_S = Number of solutions / Card[Variable Space].

Let’s imagine that, in the case of the function Figure 26, the x variable was discretized into 1000 values between x_{min} and x_{max}. The global minimum of F is obtained for all the values of x in {S} = { x_0, x_1, $[x_4, x_5]$ }. Thus, the dilution of the global minimum is

$$\mathbf{Dil_S} = \mathbf{Card[}\ \{S\}\ \mathbf{]} / \mathbf{Card[}\ \{B_F\}\ \mathbf{]}. \quad (4)$$

Where

$\mathbf{Card[}\ \{\ S\ \}\ \mathbf{]} = ((x_5 - x_4)*(x_{max} - x_{min})/1000 + 2)$.

$\mathbf{Card[}\ \{\ B_F\ \}\ \mathbf{]} = 1000$.

Dilution of the Feasible Area

Although feasible local optima are not clearly differentiated from unfeasible local optima, it is of interest to measure the size of the feasible domain with respect to the total space size. This quantity indicates the probability to find a feasible point in the domain and is expressed as

$$\mathbf{Dil_{fa}} = \mathbf{Card[}\ \{D_F\}\ \mathbf{]} / \mathbf{Card[}\ \{B_F\}\ \mathbf{]}. \quad (5)$$

Roughness

The third criterion is the Roughness of the landscape (**Rg**), which corresponds to the number of Local Optima Valleys or Hills (Figure 28), is written

Rg = Number of Valleys

For a function with no plateau, the Roughness is linked to the space size by the following equation,

$$\mathbf{Rg} * M = \mathbf{Card[}\ \{\ B_F\ \}\ \mathbf{]}. \quad (6)$$

where M is the Average of the Valleys size.

$$M = \mathbf{Mean[\ Card[}\ \{\ \text{Valleys}\ \}_i\ \mathbf{]\ ]}. \quad (7)$$

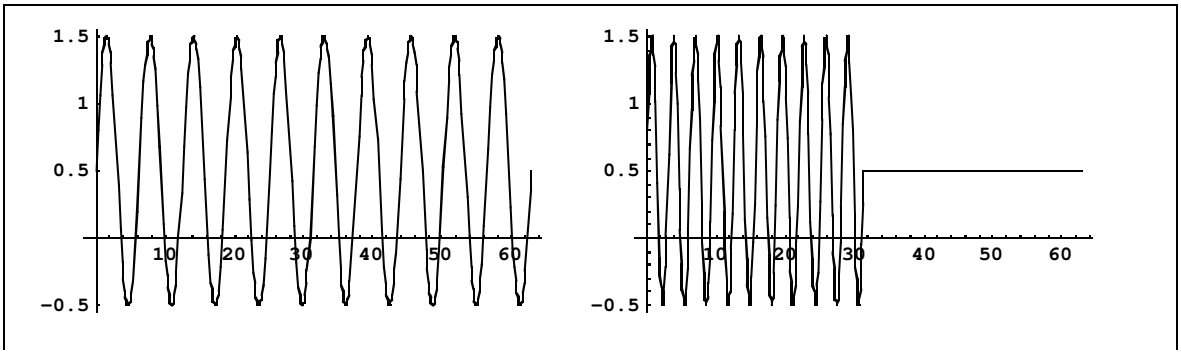

Figure 28. Example of function Roughness (Rg = 11).

The Roughness is not sufficient to characterize a landscape since the valleys can be very small and located on a very small area of the landscape. Hence a third criterion is necessary to make a difference between these cases.

Trend

Eventually, the landscape complexity might also be characterized by the existence (or the absence) of clues pointing toward the location of the solution when comparing sampled points. The factor called **Trend** (**Tr**) measures the number of clues whatever the values of the Dilution and of the Roughness are. This criterion could be compared to the number of hay twigs in the haystack pointing toward the needle.

The Trend is an integral characteristic of the function landscape. An intuitive way to define this factor is to look at the landscape from far away. The eye then erases all the local perturbations and only catches the global evolution. A translation of this characteristic is made through the definition of the influence of one point (x_0) on the other points of $\{B_F\}$ (Figure 29). In the case of the minimization of a function F which global minimum is x_{goal} and supposing that the method is able to compare previous explored points to the current one, x_0, (like a human decision maker), x_0 when compared to points with higher values, has the property to "pull" the interest of the seeker from these higher points toward itself and further in this direction. If these higher points are further from the Goal than x_0 then x_0 influence is good (Figure 30b). An external decision-maker will put his attention closer to x_0 and thus closer to the goal. On the contrary, if the points are closer to the goal than x_0 then x_0 influence on the decision maker is bad (Figure 30a). For all the points x_i having an objective value smaller than $F[x_0]$, x_0 influence is null (points x_3 and x_6 in Figure 29). The sum of all

the information available points out a location for searching the minimum of the function (Figure 30c).

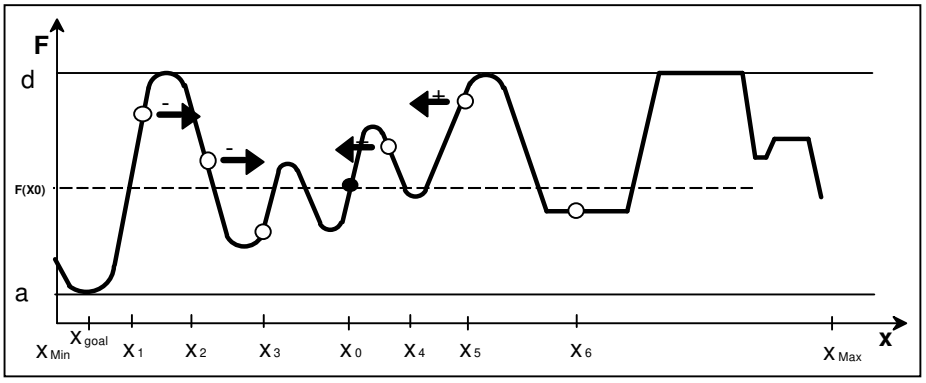

Figure 29. Trend definition. In this example the goal is to find the value of the variable x for which F is minimum (x_{goal}). The information about the position of x_{goal} depends on what previous points are considered to draw a conclusion. The point x_0 is the point analyzed and is compared to previously known points (x_1, x_2, x_3, x_4, x_5, x_6). In some cases the comparison points out a good direction (+) in some other cases it points out wrong directions (-). The sum of the number of good clues (+) and bad clues (-) gives the point Trend (**TrPoint**).

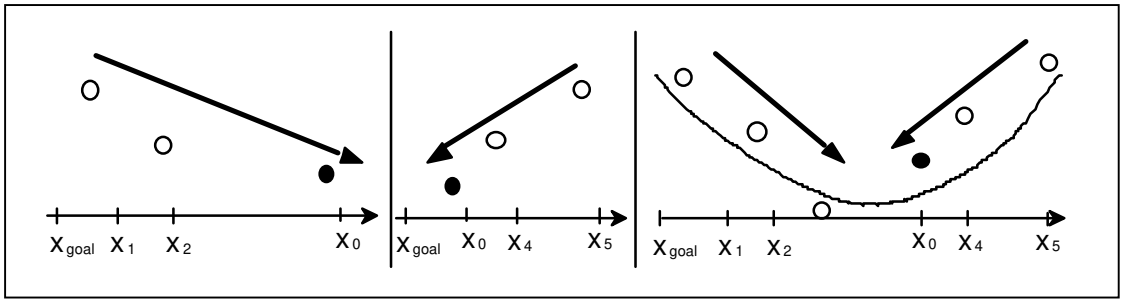

Figure 30a, b and c. Trend explanation. If the optimum of the function is Xgoal, comparing different sets of points indicates good or wrong directions. The point x_0, when compared to the points x_4 and x_5 alone, gives the information that the Optimum should be in the direction of decreasing x. When compared to x_1 and x_2 it points out that the optimum should be toward the increasing x direction. When compared to x_1, x_2, x_4 and x_5, the minimum is located between x_2 and x_0.

The Trend of a function corresponds to the sum of the Trend of all the points of the domain (point Trend). In order to translate this characteristic it is proposed to use the following mathematical definition for the case of univariate functions.

$$\mathbf{Tr}[F] =$$

$$\sum_{x \text{ in } \{B_F\}} \mathbf{TrPoint}[x]/(\mathbf{Card}[\{B_F\}]*(\mathbf{Card}[\{B_F\}] - 1)/2. \quad (8)$$

$$\mathbf{TrPoint}[x_0] = \sum_{x \text{ in } \{B_F\}} (\mathbf{unitTr}[x, x_0]). \quad (9)$$

$$\mathbf{unitTr}[x, x_0, x_{goal}] =$$
$$\{ 1 \text{ if } x > x_0 \text{ or } x < x_{goal} \text{ and } F[x_0] < F[x];$$
$$-1 \text{ if } x_{goal} < x < x_0 \text{ and } F[x_0] < F[x];$$
$$0 \text{ otherwise};$$
$$\}. \quad (10)$$

Thus, Tr varies in between 0 and 1. A low Trend indicates that few clues can be gathered on the location of the global optimum. A monotonous function presents the best Trend factor since whatever the sampling in $[B_F]$, the comparison of the samples always points in the direction of the Global Optimum (Figure 31). All the monotonous functions have the same Trend, independent of their degree of non-linearity and equal to 1.

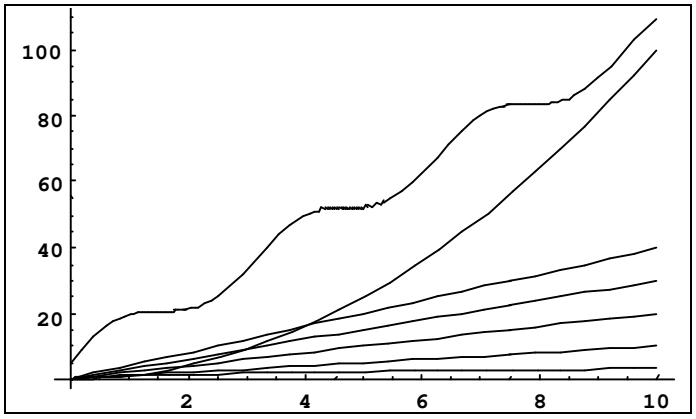

Figure 31. Trend of monotonous functions. All these functions are monotonous and thus present the same maximum Trend i.e. 1. From bottom to top the following functions are represented: Sqrt[x], x, 2*x, 3*x, 4*x, x^2, 10*x + 5*Sin[2*x] + 5.

As soon as the function becomes multimodal or non-convex, the samples taken might give wrong clues and drive the search method away from the Optimum (Figure 32).

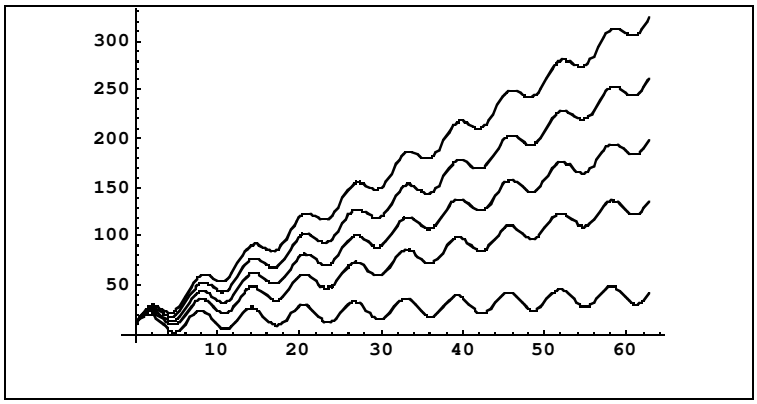

Figure 32. Trend of non-convex functions. For a function with constant Dilution and Roughness, the Trend depends on its global slope. From bottom to top, the functions presented are a*x + 10*Sin[x] + 10 for a in { 0.5, 1, 2, 3, 4, 5}. Their respective Trend are 0.57, 0.87, 0.90, 0.92 and 0.94.

Similarly to the single variable case, multivariate functions can be characterized by these three factors. The Dilution and Roughness have the same definition and the Trend can be written

$$\mathbf{Tr}[\,F\,] = \sum_{x \text{ in } BF} (\,\mathbf{TrPoint}[\,x\,]\,) \,/\, (\,\mathbf{Card}[\,\{\,B_F\,\}\,] * (\mathbf{Card}[\,\{\,B_F\,\}\,] - 1) \,/\, 2\,),$$

with

$$\mathbf{TrPoint}[\,x_0\,] = \sum_{x \text{ in } BF} (\,\mathbf{unitTr}[\,x\,,\,x_0\,]\,).$$

A method to calculate the unit Trend of a function with multiple variables and multiple Global Optima is proposed in Appendix A.

For evolved algorithms that are able to make comparisons between solutions, this factor is linked to their performance. Since the Trend is very difficult to quantify for multimodal multivariate functions, its study on one variable CDPs is used to point out the mistakes to avoid when introducing more complexity in a landscape, keeping in mind that, when the Trend is 0, the amount of information pointing in the good direction is compensated by the information pointing in another direction. This makes the deduction on the location of the Global Optimum useless. A random search (possibly combined with a local search) is as efficient as any other more sophisticated methods.

Properties

The following properties are deduced for the Dilution, the Roughness and the Trend.

- The Dilution, Roughness and Trend are constant for a given function.
- The Roughness can be increased by two types of valleys: valleys leading to Local Optima, and valleys leading to Global Optima. The first type makes the search more difficult while the second increases the chances to discover at least one optimum. From the point of view of the Trend, the first type of Valleys decreases the Trend while the second type leaves it unchanged.
- Equations 8, 9 and 10 show that the Trend is also affected by the location of the local optima. A study of this property is proposed in Appendix A.
- If [A] and [B] are two distinct subsets of [B_f] then Tr[F] on [A] U [B] = Tr[F] on A + Tr[F] on B.

Links between Landscape Complexity, Optimization Search and Probability of Success.

Given a function like F (Figure 33) of one real variable for which the size of the local valleys are the same and whose complexity is described in terms of Rg, Dil_{sol}, and Tr, it is now possible to evaluate the probability of success (Ps) of some well known optimization techniques.

Random Search

If the variable domain [B_f] is discretized into m different values, searching randomly for **Min [** F **]** over { B_f } has a probability of success equal to

Ps = **Card[** { x-Opt } **]** / **Card[** { B_f } **].**

Ps = $\mathbf{Dil_{sol}}$

Ps = 1 / m. (11)

where

m = **Card[** { B_f } **].**

Hence, m trials are required to be sure (on average) to reach the Global Optimum of F.

Randomly Restarted Local Search

In this case a Local Search method is restarted at randomly chosen points. The probability to find the best solution becomes equal to

Ps = **Card[** { Valley Optimum } **]** / **Card[** { B_f } **].**

Ps = 1 / **Rg.** (12)

On average, the better performance of this approach with respect to the previous one depends on the average and standard deviation of the size of the local valleys. If all the valleys are very small (especially the one containing the global optimum) the gain with respect to the random search is small. On the contrary if all the valleys have the same size the new probability can be calculated with,

Ps = M / m. (13)

where M is the valley size.

Evolutionary Techniques

Unlike the previous two methods whose performances were linked only to the Roughness and to the Dilution of the solution, the performance of evolutionary techniques is also linked to the Trend factor. However, only qualitative results of this dependence can be given for the moment.

The advantage of the first two methods over a more sophisticated learning search lies in the fact that their efficiency does not depend on the landscape Trend, because they do not use comparison of results to progress. For more sophisticated techniques like evolutionary algorithms (like GAs) it is impossible to predict the efficiency of the optimization process without additional information about the objective function.

Application

Several well-known functions were proposed to test the evolutionary techniques such as GA among which

$F_p[\mathbf{x}, \mathbf{x_0}] = \sum_{i=1\ldots p} x_i^2 / 40000 + 1 - \prod_{i=1\ldots p} \mathrm{Cos}[x_i / \mathrm{Sqrt}[i]])$

proposed by Griewank, and

$F[\mathbf{x}, \mathbf{x_0}] = x_1^2 + x_2^2 - \mathrm{Cos}[18 * x_1] - \mathrm{Cos}[18 * x_1]$.

proposed by Rastrigin.

Two major criticisms can be made about these functions. First, it is difficult to determine for which value of the variables they become unimodal and second, their optimum lies at the same value for all the variables (i.e. $x_1 = 0$, $x_2 = 0$, for Rastrigin's function) which introduces a pattern in the genome used by a Genetic Algorithm.

In order to remove these drawbacks a simpler function, for which the Roughness, Trend and Dilution are customizable, is built in order to serve as example and reference for measuring the performance of an intelligent minimization algorithm (Figure 33).

$F_p\ [a, \mathbf{x}, \mathbf{x_0}] = \sum_{i=1\ldots p} (a*Abs[x_i - x_{0i}] - 10*Cos[\ (x_i - x_{0i})/Sqrt[i]\ ] + 10)$

with

$-2*\Pi*n < x_i < 2*\Pi*n$ for all i

$-2*\Pi*n < x_{0i} < 2*\Pi*n$ for all i.

For p =1 :

Variables : $[\ B_F\ ] = -[2*\Pi*n, 2*\Pi*n]$ with n = 1000.

Discretization : { BF } = Disc[[x] , m].

Cardinality : Card[{ BF }] = m with m = 230.

Rg = n = 1000.

Tr = (see Table A-1 in Appendix A).

Dilsol = $1/m = 2^{-30}$.

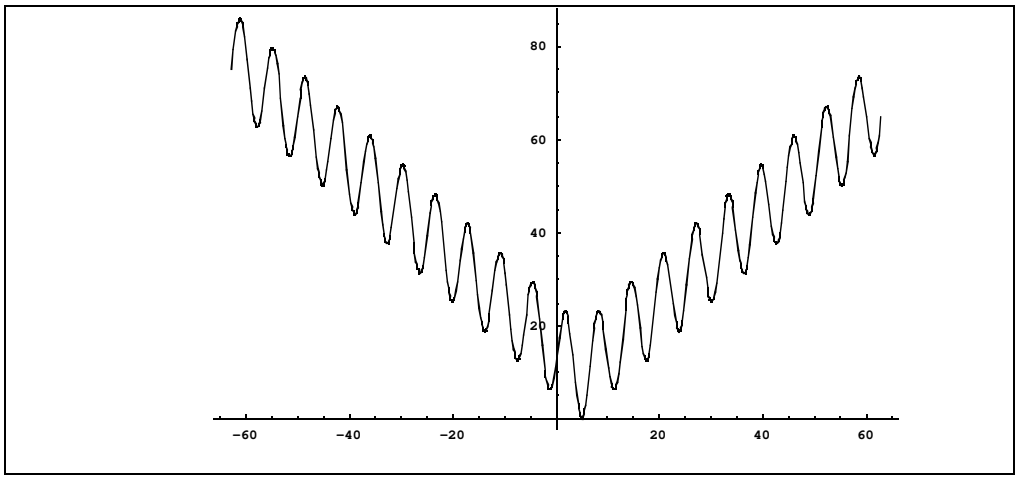

Figure 33. Function F[x] = a*Abs[x - 5] - A*Cos[x - 5] + A is plot for x in [-2nΠ, 2nΠ] and A = 10, n = 10.

The Trend of the Fp functions family increases regularly as the slope increases until the function becomes unimodal i.e. the local valleys disappear (Figure 34). This happens when the global slope (a) of the curve increases further than a limit value (see Appendix A for the calculation of this limit and for numerical values) and the Trend becomes equal to one. This limit value is a function of the parameter a, and of the amplitude of the wavelets (A).

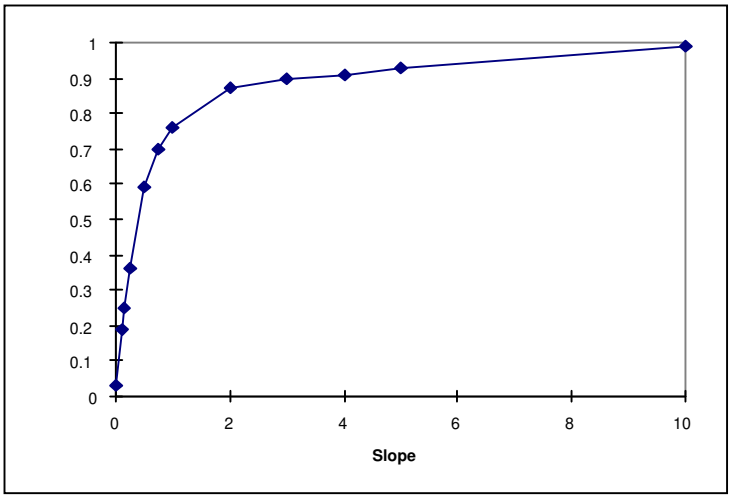

Figure 34. Evolution of the Trend of Function F when the slope 'a' increases.

The function F is then submitted to a Genetic Algorithm whose goal is to find its global minimum. The GA stops upon convergence. The performance is plotted in Figure 35 and technical parameters are given in Appendix A.

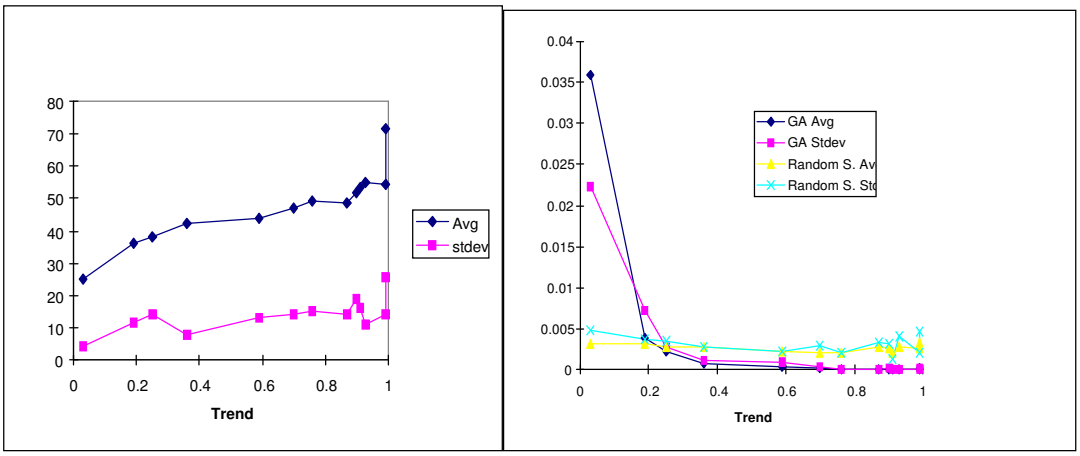

Figure 35. Evolution of a Simple Genetic Algorithm performance in function of the Trend.

As the Trend increases the GA gets more information to converge toward the global optimum. The precision levels off around 0.0004. At this stage the GA has reached the local valley of the Global optimum and thus whatever the value of the Trend beyond 0.4 the algorithm cannot perform better. Numerical results obtained from this test are given in Appendix A. The performance of a Random Search is also plotted on Figure 35 (right). The number of function evaluations allocated to this method was equal to the average number of function evaluations that the GA took to reach convergence (860). It must be noted that the GA performs worse than the Random Search for Trends below 0.2. This is explained by the fact that the GA spent much less than 50 generations to reach convergence and thus has less than 860 trials to converge toward a solution.

Configuration Design Problem Analysis

In this section the focus is put on the analysis of the CDPs by studying their basic components (objects, states/behaviors, and connectivity) their link with the optimization problem formulation using penalty functions and how they relate to the mathematical characteristics of the landscape (Dil, Tr, Rg). It must be kept in mind that, for many engineering problems, finding a feasible mechanism is more important than finding an optimal one. Hence, the following analysis involves the study of the feasible area as well as the optimum of the problem. It also intends to compare the classifications of these problems when considered as discrete and when considered as continuous. The conclusion of this study

helps to identify the difficulties to overcome in order to provide a general approach for solving families of ECDPs (i.e. involving the same objective functions).

In mechanics, as well as in CAD, there is a major difference between the object and the system. The object is an atomic solid that cannot be taken apart without being broken; whereas the system is an aggregation of objects that can move with respect to each other's. It is proposed to maintain this difference in the mathematical models involved in the CDPs description (in opposition with other approaches that tend to describe all the relations (defining the geometry of the objects and the characteristics of the assembly at the same time). Thus, on one hand, a description of the objects as indivisible whole is provided, and on the other hand, a description for the system as a relative or global positioning of the objects is used. The characteristics of interest in this study are system level characteristics like volume, inertia, and maintainability.

Finally, in order to get quantitative results, the variable spaces under study are discretized such that the precision of the placement will be, at first, 1/1000 of the total system size (in the implementation, the results obtained will have at least this precision). This choice allows the capture of local valleys as small as 2/1000 of the system size and to compare the efficiency of the CDOM independently from the size of the system. This precision is usually sufficient for most mechanical engineering systems, however it can be changed at will.

Objects, Variables and Variable Space

The first step for calculating the Dilution consists of obtaining the size of the variable space for each CDP. The literature review pointed out few different object types that are grouped here according to the smoothness of their shape which gives birth to different types of CDPs and different techniques to solve them.

- squares, cubes, triangles and other polygonal objects with few faces
- smooth objects
- sphere like
- freeform and polygonal objects with a large number of faces.

The analysis of the various methods used to solve packing problems showed that the type of algorithm used matched these categories. Optimizing the placement of the first category of elements is usually done by using branch and bound techniques or other

combinatorial heuristics (combinatorial optimization) while problems involving the elements of the second category are solved using optimization methods for continuous functions.

In both cases, and given a Cartesian coordinate system, each object can be positioned, by 3 translation variables (x, y, z) with respect to the origin of the coordinate system and three rotation angles (rx, ry, rz) with respect to a particular point (object center of gravity for example). Squares, triangles, polygons, boxes or polyhedrons, were usually involved in problems in which their positions were restricted. First, they had to touch each other, and second, they presented symmetrical shapes. Hence, the Variable Space for these cases was dramatically reduced.

Let's compare well known CDPs presented in the literature review as if their variables were continuous. It is reminded, that for each problem, the variable space is an hyperbox [B_F] whose general definition is

$$[\ B_F\ ] = \left(\prod_{i = 1\ ..\ Nvar} [\ x_i\ ]\right)^{N\_obj}$$

and its discrete version is noted,

$$\{\ B_F\ \} = \left(\prod_{i = 1\ ..\ Nvar} \mathbf{Disc}\ [\ [\ x_i\ ],\ m_i\ ]\right)^{N\_obj}.$$

The cardinal of this last set is

$$\mathbf{Card}\ [\ \{\ B_F\ \}\ ] = \left(\prod_{i = 1\ ..\ Nvar} m_i\ ]\right)^{N\_obj}$$

where N_obj is the number of components of the system and the discretizations (m_i) are chosen equal to 1000.

According to this definition the only difference that might change the size of the variable space from one CDP to the other is the number of objects and the number of variables per object. The number of variables is thus the only parameter to consider to know the size of the variable space (due to the fact that the number of values that a variable can take is fixed).

CDP 1 and 1": Squares and Cubes Packing

First, let's examine the tricky case of packing squares, (Figure 16). All the objects have the same shape, symmetrical by 90 degrees rotations. For all the instances of this CDP

involving a number of N squares such that Sqrt[N] is an integer number, the rotation variables can be removed from the problem.

variables : (x, y, 0) and (0, 0, 0)

$[B_F] = ([x] \underline{*} [y])^{\underline{N\ squares}}$

If the variable axes are discretized such that each variable can take 1000 different values (i.e. the objects inside a 1 square meter system are then positioned with a precision of 1 millimeter), hoping that solutions will not be missed by this choice, the total space size becomes

$\{ B_F \} = (\mathbf{Disc}[[x] , 1000] \underline{*} \mathbf{Disc}[[y], 1000])^{\underline{N\ squares}}$

$\mathbf{Card}[\{ B_F \}] = (1000 * 1000)^{N\_squares}$

$= (1x10^{6})^{N\_squares}.$

However this simplification supposes that the solution was foreseen. In order not to miss solutions for all the instances of this problem we have to authorize the squares to rotate. By looking at well-known solutions, a simplification seems to appear: the restriction of the rotations to 2 values, 0 and 45 degrees. Hence, the characteristics of this CDP are

variables : (x, y, 0) and (0, 0, rz)

$[B_F] = ([x] \underline{*} [y] \underline{*} [rz])^{\underline{N\ squares}}$

$\{ B_F \} = (\mathbf{Disc}[[x] , 1000] \underline{*} \mathbf{Disc}[[y], 1000] \underline{*} \{0,45\})^{\underline{N\ squares}}$

$\mathbf{Card}[\{ B_F \}] = (1000 * 1000 * 2)^{N\_squares}$

$= (2x10^{6})^{N\_squares}.$

Proving that 0 and 45 degrees are the only possible angles seems difficult. Hence it would be wiser to extend the rotation range from 0 to 45 degrees continuously.

$\mathbf{Card}[\{ B_F \}] = (1000 * 1000 * 1000)^{N\_squares}$

$= (1x10^{9})^{N\_squares}$

Three additional variables must be added for the cubes in order to take into account the third dimension.

variables : (x, y, z) and (rx, ry, rz)

$[B_F] = ([x] \underline{*} [y] \underline{*} [z] \underline{*} [rx] \underline{*} [ry] \underline{*} [rz])^{\underline{N\_cubes}}$

$\{ B_F \}$ = **Disc** [[x] ,1000] $\underline{*}$ **Disc**[[y], 1000] $\underline{*}$ **Disc**[[z], 1000] $\underline{*}$ {0,45} $\underline{*}$ {0,45} $\underline{*}$ {0,45})$^{\underline{N\_cubes}}$

Card[$\{ B_F \}$] = $(1000 * 1000 * 1000 * 2 * 2 * 2)^{N\_cubes}$

= $(8*10^9)^{N\_cubes}$

And for its continuous version:

Card[$\{ B_F \}$] = $(1000 * 1000 * 1000 * 1000 * 1000 * 1000)^{N\_cubes}$

= $(1*10^{18})^{N\_cubes}$

CDP 2 and 2": Pallet Loading

A simpler example is the pallet loading problem (CDP_2), for which the aim is to load a rectangular carrier with rectangular pallets that can be different in weight (A typical pallet is 1*1.2*H (meters) in size and a typical load carrier, like a boat container is L: 12m W: 2.34m H: 2.38m) (Figure 18). The goal is to put the center of gravity of the load as close as possible to a target point. The pallets can only be rotated by 90 degrees because of loading and unloading constraints. The total space size is :

variables : (x, y, 0) and (0, 0, rz)

$[B_F] = ([x] \underline{*} [y] \underline{*} [rz])^{\underline{N\_obj}}$

$\{ B_F \}$ = (**Disc** [[x], 1000] $\underline{*}$ **Disc**[[y], 1000] $\underline{*}$ {0,90})$^{\underline{N\_obj}}$

Card [$\{ B_F \}$] = $(1000 * 1000 * 2)^{N\_Obj}$

= $(2*10^6)^{N\_Obj}$

When the rotation is made continuous,

Card [$\{ B_F \}$] = $(1000 * 1000 * 1000)^{N\_Obj}$

= $(1*10^9)^{N\_Obj}$

A relative positioning strategy used by Grignon and Fadel (Grignon and Fadel 1995), among other advantages, changed this size to

Card [{ B_F }] = N_obj! * ($2*10^2$)$^{N\_Obj}$ (discrete rotation)

Card [{ B_F }] = N_obj! * ($1*10^3$)$^{N\_Obj}$ (continuous rotation)

and thus was advantageous for a number of objects that depends on the discretization. If all the pallets are the same, this number is reduced to

Card [{ B_F }] = ($2*10^2$)$^{N\_Obj}$

The cardinality of the 3D pallet loading CDP is

{ B_F } = (**Disc**[[x], 1000] * **Disc**[[y], 1000] * **Disc**[[z], 1000] * {0,90})$^{N\_obj}$

Card [{ B_F }] = (1000 * 1000 * 1000 * 2)$^{N\_Obj}$

= $(2*10^9)^{N\_Obj}$

which gives in its extended version,

Card [{ B_F }] = (1000 * 1000 * 1000 * 1000)$^{N\_Obj}$

= $(1*10^{12})^{N\_Obj}$.

With a relative positioning

Card [{ B_F }] = N_obj! * ($2x10^3$)$^{N\_Obj}$ (discrete rotations)

Card [{ B_F }] = N_obj! * ($1x10^4$)$^{N\_Obj}$ (continuous rotations)

If all the pallets are the same this number is reduced to :

Card [{ B_F }] = ($2x10^3$)$^{N\_Obj}$

CDP 3 and 3" : Circles and Spheres packing

The circles and spheres CDPs (CDP_3, CDP_3") (Figure 20 and 21) respectively are problems involving only translations.

variables : (x, y, 0) and (0, 0, 0)

$[B_F] = ([x] \underline{*} [y])^{\underline{N\_obj}}$

$\{ B_F \} = (\mathbf{Disc} [[x], 1000] \underline{*} \mathbf{Disc}[[y], 1000])^{\underline{N\_obj}}$

$\mathbf{Card} [\{ B_F \}] = (1000 * 1000)^{N\_Obj}$
$= (1x10^6)^{N\_Obj}$

variables : (x, y, z) and (0, 0, 0)

$[B_F] = ([x] \underline{*} [y] \underline{*} [z])^{\underline{N\_obj}}$

$\{ B_F \} = (\mathbf{Disc} [[x], 1000] \underline{*} \mathbf{Disc}[[y], 1000] \underline{*} \mathbf{Disc}[[z], 1000])^{\underline{N\_obj}}$

$\mathbf{Card} [\{ B_F \}] = (1000 * 1000 * 1000)^{N\_Obj}$
$= (1*10^9)^{N\_Obj}$

CDP 4 and 4": 2D and 3D Pentominos

A fourth example is the 2 dimensional Pentominos (CDP_4) (Figure 11). Each of the 12 objects can translate in the X and Y direction on a 8x8 board and rotate by 90 degrees around a vertical axis. The translation is not continuous since the solutions require that the translation is discrete thus restricting the possible vectors to 8 integer values on each axis. Further simplification could be done if a (complex) heuristic is developed in order not to allow the objects to overlap. However, if the problem remains according to the previous formulation, the total variable space size is:

variables : (x, y, 0) and (0, 0, rz)

$\{ B_F \} = (\{0,..,7\} \underline{*} \{0,..,7\} \underline{*} \{0, 90, 180 ,270\})^{\underline{12}}$

$\mathbf{Card} [\{ B_F \}] = (8 * 8* 4)^{12}$

$= 8*10^{28}$

and the variable space size of its continuous version is

$\mathbf{Card} [\{ B_F \}] = (1000 * 1000 * 1000)^{12}$

$= (1*10^9)^{12}$

The 3D Pentominos are played in a 3x4x5 box. For each object, the authorized displacement vectors are T(x, y, z), R(rx, ry, rz). The translations take discrete integer values respectively in the sets {0,1,2}, {0,1,2,3} and {0,1,2,3,4}. The rotations are also discrete integer and can take 4 different values. Each angle takes its values in the set {0, 90, 180, 270}. Hence, the cardinality of this CDP is :

variables : (x, y, z) and (rx, ry, rz)

$\{ B_F \} = (\{0,..,2\} \underline{*} \{0,..,3\} \underline{*} \{0,..,4\} \underline{*} \{0,90,180,270\} \underline{*} \{0,90,180,270\} \underline{*} \{0,90,180,270\})^{\underline{12}}$

Card $[\{ B_F \}] = (3 * 4 * 5 * 4 * 4 * 4)^{12}$

$= 1x10^{43})$

or

Card $[\{ B_F \}] = (1000 * 1000 * 1000 * 1000 * 1000 * 1000)^{12}$

$= (1x10^{18})^{12}$

for the continuous case.

Note : The 2D pentominos packing problem has the additional symmetry variable since all the components can be flipped upside down. This variable was removed from the analysis since the problem already presents several solutions just considering rotations and translations. Moreover considering only translations and rotations makes this CDP very close to the mechanical applications, while introducing symmetries implies that for each component two shapes are possible which is against the assumptions of this work.

This problem does not occur for 3D pentominos since the components are 2D components extended by extrusion along the third direction which makes all the possible symmetries accessible via rotations.

CDP 5 and 5": Freeform 2D and 3D cases

Final and more general examples are the freeform 2D and 3D CDPs (CDP_5, CDP_5") in which each object has two translations and one rotation in the 2D case and three translations and three rotations in the 3D case. If we discretize each translation in 1000 values and each rotations in 360 values the cardinality for these CDPs are:

variables : (x, y, 0) and (0, 0, rz)

$\{ B_F \} = (\mathbf{Disc} [[x], 1000] \underline{*} \mathbf{Disc}[[y], 1000] \underline{*} \mathbf{Disc}[[rz], 360])^{\underline{N\_obj}}$

Card $[\{ B_F \}] = (1000 * 1000 * 1000)^{N\_obj}$

$= (1*10^{9})^{N\_obj}$

variables : (x, y, z) and (rx, ry ,rz)

$\{ B_F \}$ = (**Disc** [[x], 1000] ***** **Disc**[[y], 1000] ***** **Disc**[[z], 1000] ***** **Disc**[[rx], 360] ***** **Disc**[[ry], 360] ***** **Disc**[[rz], 360])$^{N\_obj}$

Card [$\{ B_F \}$] = $(1000 * 1000 * 1000 * 1000 * 1000 * 1000)^{N\_obj}$

$= (1*10^{18})^{N\_obj}$

CDPs Variable Space Size Summary

Now let's compare all these examples. The Cardinal of the variable space is usually a good measure of the probability to randomly find a solution since the number of solutions to these problems is small in comparison to the space size (Table III). Since the Pentominos problem involves 12 objects let's compare 12 objects CDPs for which space sizes are regrouped in Table IV.

Table III. Total variable space size for N objects discrete and continuous CDPs.

		2D		3D
Cardinality	CDP_1	$(1x10^{6})^{N}$ ->$(1x10^{9})^{N}$	CDP_1"	$(1x10^{9})^{N}$->$(1x10^{9})^{N}$
	CDP_2	$(2x10^{6})^{N}$->$(1x10^{9})^{N}$	CDP_2"	$(2x10^{9})^{N}$->$(1x10^{9})^{N}$
		$N!(2x10^{2})^{N}$->$(N!(1x10^{3})^{N}$		$N!(2x10^{2})^{N}$->$N!(1x10^{5})^{N}$
	CDP_3	$(1x10^{6})^{N}$	CDP_3"	$(1x10^{9})^{N}$
	CDP_4	$8x10^{28}$->$(1x10^{9})^{12}$	CDP_4"	$1x10^{43}$->$(1x10^{18})^{12}$
	CDP_5	$(3.6x10^{8})^{N}$->$(1x10^{9})^{N}$	CDP_5"	$(4.67x10^{16})^{N}$->$(1x10^{18})^{N}$

Table IV. Total variable space size for 12 objects discrete and continuous CDPs.

		2D		3D
Cardinality (Log)	CDP_1	75->108	CDP_1"	108->216
	CDP_2	76->108	CDP_2"	112->216
		36->68		48->80
	CDP_3	72	CDP_3"	108
	CDP_4	29->108	CDP_4"	43->216
	CDP_5	103->108	CDP_5"	200->216

Several conclusions can be drawn from this study.

- While the size of discrete CDPs relies on the Cardinality of each variable range, the size of the continuous CDPs, assuming a constant precision in variable space,

depends on the number of variables. This has for consequence to eliminate the differences between the CDPs.

- An exhaustive search of the discrete CDPs variables space is out of reach of today's computer capabilities. Continuous CDPs are even worse than their discrete siblings. Therefore, some knowledge must be introduced either in the formulation, or in the resolution method or in both in order to obtain a useful solution in a reasonable time.

- The use of relative placement is an example of new formulation reducing space size. The size of CDP_2 is reduced by several orders of magnitude with this method. However, this technique requires the knowledge of how the configurations could be restricted without loosing solutions. This is an interesting strategy for mechanical systems which in general are made of components with a restricted number and well-specified degrees of freedom.

System Description, Constraints and Feasible/Unfeasible Regions

The possible interaction between objects are put into three categories that match the three types of constraints used in the CDPs mathematical formulation.

- Physical impossibilities (Inequality constraints G_i)
- Mechanical functional links (Equality constraints H_i)
- Behavioral mechanical constraints (Min and Max constraints)

Physical Impossibilities

Physical impossibilities constraints express the implicit assumption made when manipulating physical objects in real life, which is no longer true as soon as these objects becomes virtual like in a CAD environment, i.e. the impossibility to get an interference between two objects. The mathematical translation of this impossibility is made through the use of inequalities specifying that any part of any object must stay outside another object. Whenever this is possible, these constraints are removed from the problem by a clever formulation that ensures that no object will ever overlap whatever their position. This kind of formulation was used in CDP_2 and, in addition to the smaller variable space size, gives the advantage to speed up the evaluation of each solution since no interference checking is made between the components of the system. However, this 'clever' formulation is not possible for all CDPs (especially for freeform CDPs), then interference must be checked.

In order to assure the consistency in the way interference is handled, penalty functions are introduced in the optimization problem formulation. Among the different types of penalties, the one proportional to the amount of overlap between the objects is used in the following. In the example shown on Figure 36, a pallet can roll along a linear path (the x-

axis). The goal of this CDP is to place the pallet such that it does not overlap with the others which have fixed positions.

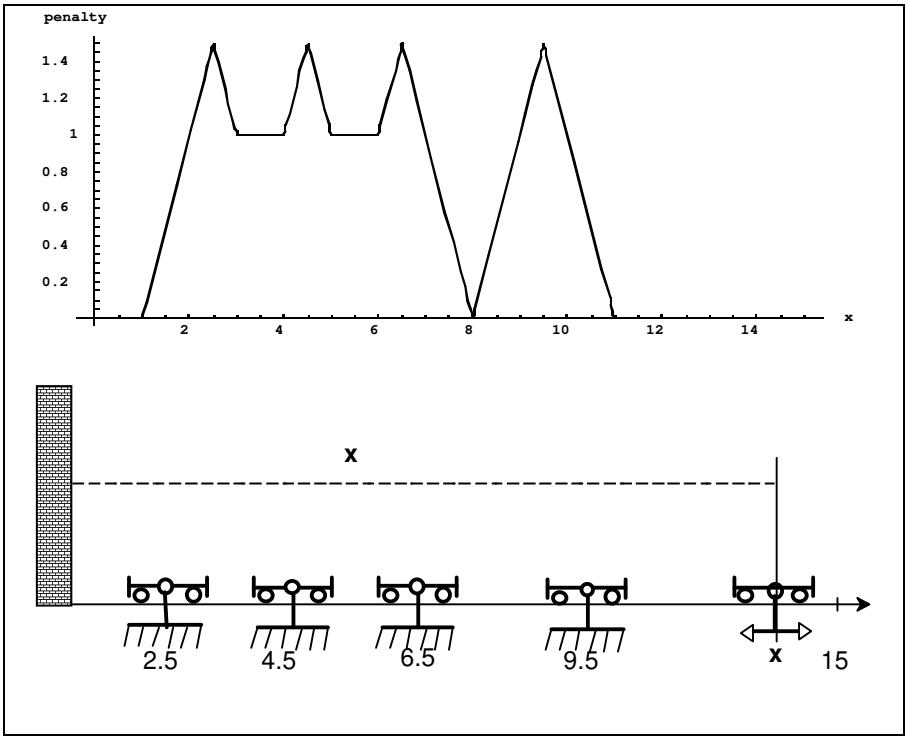

Figure 36. Overlapping penalty function. In this example, overlapping between the moving pallet (x) and the fixed pallets is penalized by a linear function of the amount of overlap of the two pallets.

Each time an overlap occurs, the penalty function increases proportionally to the amount of overlap. Hence, from a mathematical point of view, finding a feasible position consists of finding x such that the penalty is minimum. Several solutions exist and some are more difficult to find than others. For the first time, the notion of a CDP landscape, introduced in the first part of this chapter, appears, for which the Dilution, Roughness and Trend can be studied.

Roughness Due to Inequality Constraints

Penalty functions have for consequence an increase in the Roughness of the landscape by an amount which is a function of the number of intersecting objects and of their size with respect to the system size. Exactly evaluating $\mathbf{Rg}_{\mathbf{pen}}$ depends on the possibility of

interference of the components with each other's and thus on an a priori knowledge of the design variable space. Without this knowledge, the Roughness introduced by the penalty term into each landscape can be evaluated in the best, worst and average cases by assuming that some components can move with respect to an absolute coordinate system and thus can interact with all the other components (static or dynamic); while some other components can only move with respect to a relative reference (like another component). Hence, if a CDP involves 'a' objects which can move all over the 3D space, 'r' objects which can move with respect to some of the previous objects and 's' static objects, the number of two by two interference is :

if r > 1

$$N_{int} = r(a-1) + \frac{a(a-1)}{2} + a*s + \frac{(r-2)(r-3)}{2} + r*s \qquad (14)$$

if r < 1

$$N_{int} = r(a-1) + \frac{a(a-1)}{2} + a*s + r*s \qquad (15)$$

Then, the Roughness of a CDP can be approximated by the following equations :

when r = 0,

$$Rg = (1+s)^a - \frac{a(a-1)}{2}\left[(1+s)^{a-1} - 1 - s\right] + a(a-1)\left[(1+s)^{a-1} - 1 - s\right] - (1+s) + (1+s)a!$$

otherwise,

$$Rg = (1+s)^{a+r-2} - \frac{(a+r-2)*(a-1+r-2)}{2}\left[(1+s)^{a-1+r-2} - 1 - s\right] + (a+r-2)(a-1+r-2)\left[(1+s)^{a-1+r-2} - 1 - s\right] - (1+s) + (1+s)(a+r-2)! \qquad (16)$$

The procedure to obtain this last expression is detailed in Appendix A.

While the number of interactions between the objects is growing like a polynomial function, the Roughness is exponential. This is the expression, for continuous CDPs, of what is the NP nature of the Discrete CDPs.

The application of Equations 14, 15 and 16 gives the exact number of local valleys and Roughness to be expected for the CDP described in Figure 36 i.e. 4 and 5. The landscape

provided with a case in which two pallets can move along the x-axis and three are fixed is described in Figure 37. In this last case s = 3 and a=2. Hence, the number of interactions is 7 and the Roughness is 20.

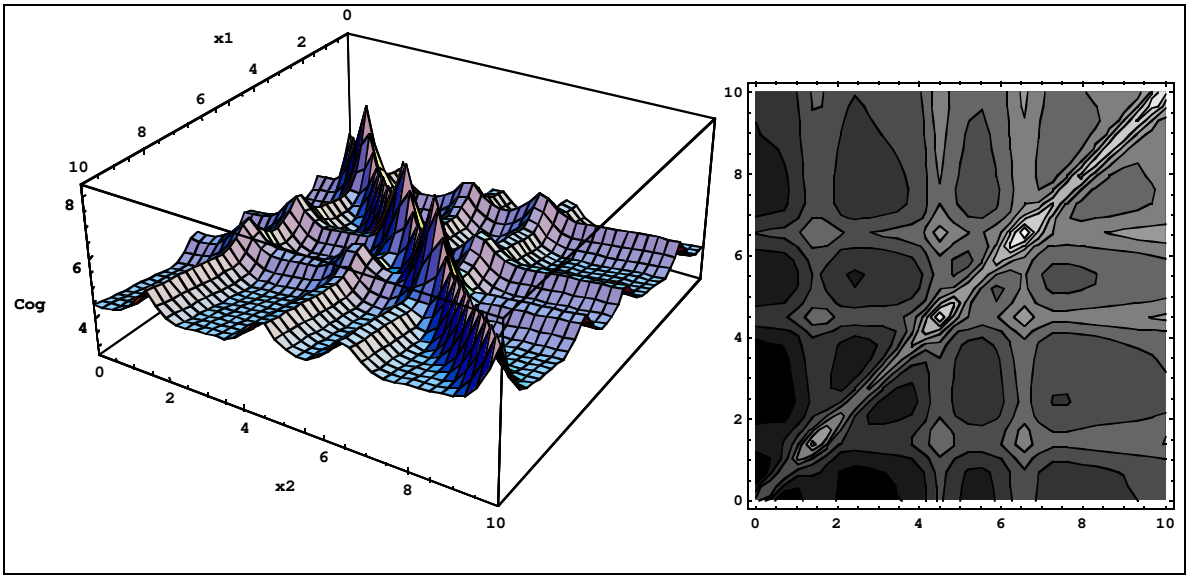

Figure 37. Landscape of a 5 pallet 1D CDP in which 3 pallets are fixed.

In order to rate the 5 CDPs chosen in this chapter their Roughness and number of interferences are calculated using equations 14 and 15 (Table V).

Table V. Worst case roughness with 12 objects CDPs.

		2D		3D
Rg_{pen} (Log)	CDP_1	20	CDP_1"	20
	CDP_1'	20	CDP_1'"	20
	CDP_2	17.6	CDP_2"	17.6
	CDP_3	20	CDP_3"	20
	CDP_4	20	CDP_4"	20
	CDP_5	20	CDP_5"	20

The total variable space size (Table IV) is closely related to the combinatorial complexity of discrete problems. It can be considered as a worst case complexity for

Continuous CDPs. However, it is believed in this work that the continuous nature of ECDPs makes them much easier to solve than their discrete counter parts, hence, Roughness (Table V) is preferred for measuring CDPs complexity.

Feasible Area Dilution

The Feasible Area is the domain defined by all the points of the variable space that do not violate any constraints. The second characteristic of the landscape that is of interest for this work is the relative size of this domain with respect to the total variable space size. The dilution of this feasible area corresponds to the ratio of the size of the feasible area over the size of the total variable space. The bounds of this Dilution can be defined as follows. Assuming that a component can be placed everywhere in the system except on the location occupied by the components already inside, the upper bound can be defined as

$$\prod_{i=0..N} (s - i*os) / s \qquad (17)$$

where

os = Minimum Object Size + half its size (Surface or volume),

s = System Size (Surface or Volume).

N = Number of objects.

The lower bound can also be defined if, the minimum objects size is replaced by the maximum object size in equation (17).

These bounds can be refined if instead of considering the minimum or maximum of the sizes of all the objects for the variable ‘os’, each object size is introduced individually in equation (17).

The study of the evolution of these bounds when the number of objects increases shows that the chance of placing a given number of components in one shot, without any interference, decreases quickly (Figure 38). This is corroborated by the application of equation (17) to the 5 test cases (Table VI).

Table VI. Minimum Dilution of the feasible domain for the 12 objects CDPs. The dilution of the freeform CDP (CDP_5) needs additional precision to be determined (size of the objects and size of the system).

	2D		3D	
$\mathbf{Dil_{FA}}$	CDP_1	0.007	CDP_1"	0.06
	CDP_2	0.036	CDP_2"	0.28
	CDP_3	0.0004	CDP_3"	0.19
	CDP_4	-> 0	CDP_4"	-> 0
	CDP_5	?	CDP_5"	?

It is clear, that in the cases presented so far, the search will be unable to produce a feasible solution just by chance. Hence, it seems likely that the search method will investigate more unfeasible points than feasible ones. Constraints seem to be the sore point of the numerical algorithms applied to continuous CDPs since a user might accept a local optimum for the objective but he would surely reject any unfeasible solutions.

The Feasible area size is reduced by two factors, first if the components grow in size with respect to the system size (the average object size increases), second when the number of components increases. For systems involving the same number of components, an interesting question, that can be answered by the curves plotted on Figure 34, concerns the identification of the Minimum ratio between the object and system size that presents a given percentage of feasible area (Dil_{FA}).

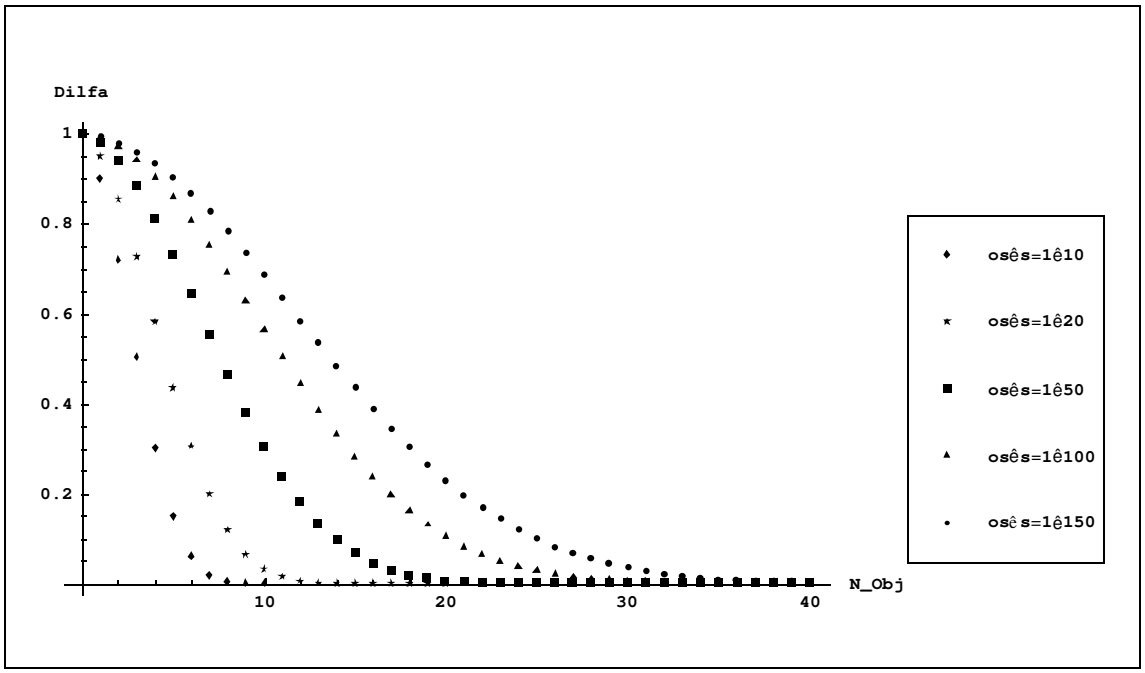

Figure 38. Evolution of the Feasible Area Dilution. The number of objects (N_Obj) is represented on the horizontal axis. Several ratios object size vs. system (os/s) are plotted.

The answer to this question helps to choose an optimization method privileging either optimum finding or constraint satisfaction.

In response to these cases, three types of strategies have emerged. A strategy which makes the system artificially bigger and which allows the components to wander off far away from each other and thus into feasible regions (no possible overlap). A second strategy consists of removing for a time the constraints due to interference. In both cases the dilution of the feasible area is artificially increased in order to give, to the search method which is able to learn, the time to gather information about the objective alone. A third strategy consists of discovering feasible areas before introducing the objective function. The system size is kept as small as possible and the search is performed first on the constraints then on the objective and the constraints. The limit cases defining which method is best suited are still undefined.

The Feasible Area Dilution is the second criterion used for characterizing the complexity of the ECDPs. One of its interesting properties is to link the objects characteristics (size) to the system characteristics. A second property than can be concluded from the study of Figure 38 is the existence of an asymptotic number of components that can

be placed inside a system when the objects size decreases (see Appendix A for an analysis of this property).

Mechanical Functional Constraints: Equality Constraints

The second type of constraints regroups the mechanical constraints. It ensures that the system is consistent with its specifications. These mechanical constraints represent the degrees of freedom of the components of the system (Figure 39). The position of each component with respect to the others is then known through the use of geometrical constraints (perpendicular, parallel, distance, angle, concentricity, fix).

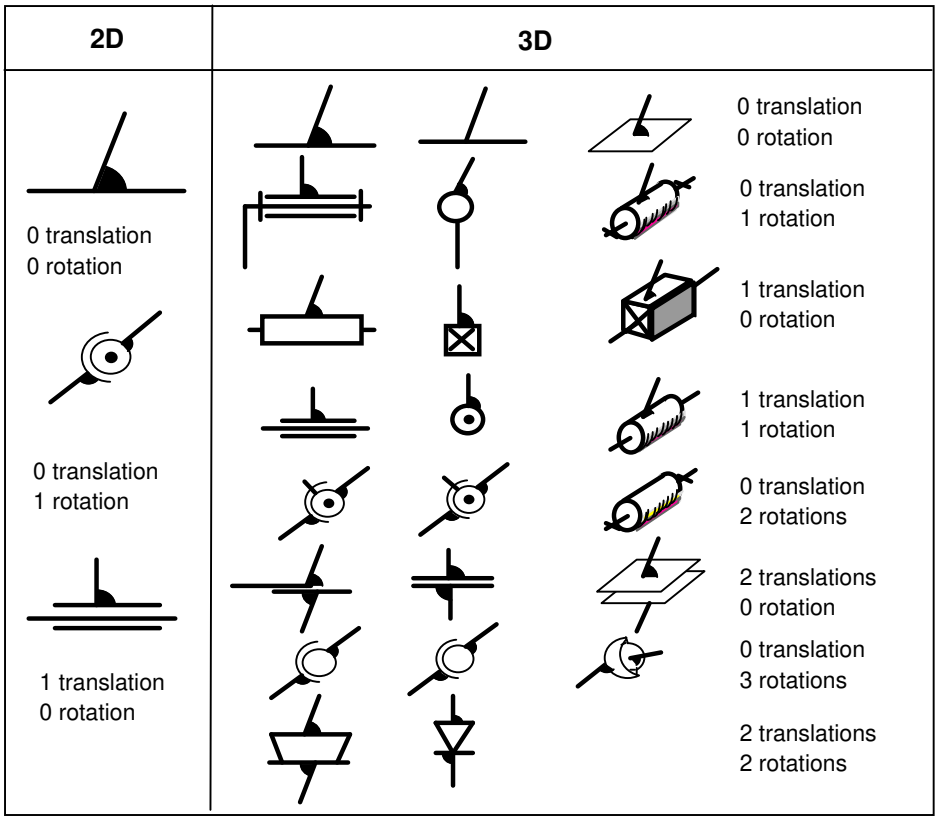

Figure 39. The Mechanical Functional Links (ISO 3956). These links describe the possible degrees of freedom that a component can have with respect to another one.

For instance, two pulleys must have their axles parallel at a certain distance from each other. They can rotate around their axles. This is translated by a fixed well-defined

position for each of the cylindrical mechanical links (Figure 40). The system cannot work if either the distance or the parallelism between the axles is changed.

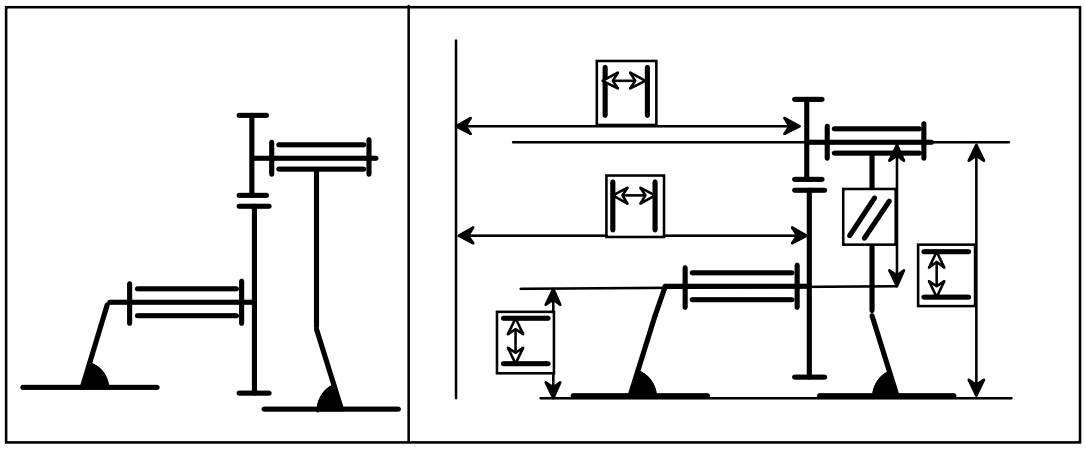

Figure 40. Example of mechanical functional links. The pulleys both have a relative position with respect to the frame they are attached to.

In CAD, these constraints are handled separately by specialized algorithms involving graph topology study and LP solving techniques as well as algebraic simplifications (D-Cubed 1996, ALS 1997). Including these additional constraints in the CDP would require to add a large number of linear and non-linear equalities. This would make the success of the search method dependent on the efficiency of all the specialized algorithms introduced. A second approach consists of including all the constraints as penalty terms like what was done for the interference. This introduces additional local optima that are difficult to quantify in terms of number and size and which split the domain $[B_F]$ into several small regions (due to the equality constraints) that are known to be difficult to handle by numerical methods.

Behavioral Mechanical Constraints

Finally, there exists a category of constraints concerning the well functioning of the system and that are obvious for the engineer, they have to be enforced at best and not exactly. This characteristic makes them look like objective functions instead of constraints. For example the engineer knows that two objects must be placed as far as possible from each

other's for security reasons. In the case of CDPs, two of these constraints were selected : as far as possible and as close as possible.

These constraints are introduced as the square of the distance between the two components involved (Figure 41).

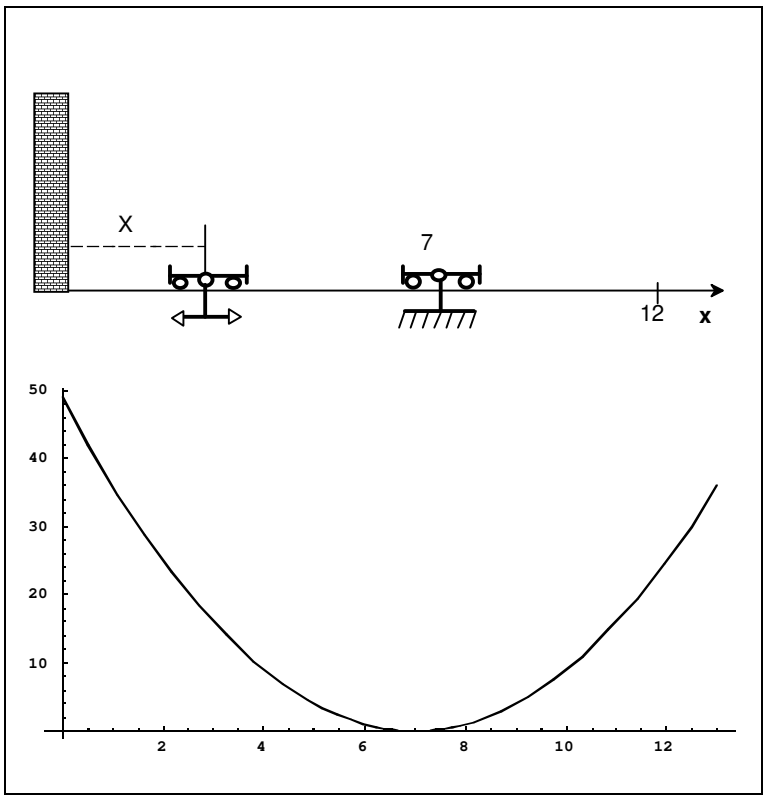

Figure 41. Constraint 'as far as possible'. This constraint is plotted for 2 objects that translate along an axis.

As was explained earlier, this unimodal function has a Trend equal to one and thus its introduction in the formulation of the CDP does not bring more difficulty to the global problem. As for the Roughness, this function has little influence compared to the high number of valleys introduced by the no-interference penalty functions.

System Description and Constraints Summary

Several conclusions emerge from this study :

- First, among the three types of constraints necessary to build mechanical assemblies it seems that the non-interference constraint and the behavioral constraints can be introduced in the generalized formulation as penalty functions. A different way to consider the functional links must be proposed by the CDOM.

- The global Roughness introduced by the no-interference constraints grows like an exponential function of the number of intersecting components. This number is the translation, for continuous CDPs, of the NP nature of their discrete counter part. Hence, it seems that, preventing objects from intersecting is a priority for the optimization of free-form CDPs.

- The solutions of CDPs with volume minimization goal lies in a region of high Roughness since objects are closely packed. This leads to small Dilution of the feasible area since it is usually easy to bound the system size with a close approximation of the solution. This is not the case for all the CDPs and thus the solutions might be found in area of low Roughness. This is an additional challenge for a method trying to tackle Multicriteria CDPs with both types of solutions.

Behavior, States, Objectives and Solutions

Until the introduction of the objective function in the formulation, the CDPs considered so far are constraint satisfaction problems. The objectives, $F[\mathbf{x}]$, of the common CDPs (puzzles, stock cutting and volume optimization with cube like objects) have several characteristics that must be pointed out. In addition to the change of Roughness of the CDP landscape (dependent on each objective function individually) they also bring what we are looking for i.e. a global optimum (in the objective space) that corresponds to one or many solutions in the variables space.

Roughness of the Reference CDPs Objective Function

Let's imagine one moment that the system components can overlap. These CDPs represent examples worst cases in term of interactions and therefore in terms of Roughness. The volume and center of gravity objective functions are unimodal functions of the translations variables. When the rotations are introduced these objectives become multimodal. If each object is encapsulated into either a circle or a sphere, the new CDP defined approximates the initial CDP and is unimodal since independent of the rotations. The Roughness of the two functions considered is equal to 1 (since there is only one valley for all the global optima corresponding to a configuration in which all the spheres have the same center). Additional solutions are introduced only because of the existence of the non-interference constraint, which makes the solution non-trivial for these two functions.

Without this approximation, the landscape of the volume and balance is changed with additional local valleys that depend on the shapes of the objects. Objects with large

variations in their dimensions, i.e. having a principal inertia matrix with large difference in its diagonal terms, can introduce large variations in the objective function landscape when they rotate around their center of gravity. However, the number of local optima introduced by the volume or by the center of gravity objective functions is inferior to the local optima due to the no-interference constraint. This is why it is chosen, to approximate the Roughness of a CDP by applying equation (16).

Solutions Dilution of the Reference CDPs Objective Functions

The case in which there are infinitely many solutions is translated into a finite number of solutions because of the use of discretized variables. A part of the difficulty to solve a CDP comes from the size of the set of all the solutions compared to the size of the total variable space to be explored (the **Dilution** of the solution).

Study of the Solution Dilution

The Dilution of a CDP solution can be bounded by

$$1 \,/\, \mathbf{Card}[\ \{B_F\}\ ] < \mathbf{Dil_{sol}} < \mathbf{Dil_{FA}}$$

The number of solutions of a CDP ($\mathbf{N\_sol_{CDP}}$) can be decomposed into three independent factors :

- **N_sol** : The number of solutions obtained if all the objects are considered different and if the final configuration has no axes of symmetries, which implies that no objects can be permuted and the final configuration cannot be translated or rotated as a whole. This number depends on the characteristics of the objective function. It is at least equal to 1 in a SOP. It is equal to the Cardinal of the Pareto set for a MOP. In both cases, this number is unknown.
- $\mathbf{N_{sh}}$: The number of solutions introduced by permutations of two objects when they are the same.
- $\mathbf{N_{sym}}$: The number of solutions obtained when the domain is symmetrical or when a configuration solution of the problem can be moved as a rigid body (in translation and rotation) without changing the objective value.

The solution dilution then becomes

$$Dil_{sol} = N\_sol_{CDP} \,/\, Card\ [\ \{\ B_F\ \}\ ]$$

$$Dil_{sol} = (\ N\_sol \times N_{sh} \times N_{sym}\ ) \,/\, Card\ [\ \{\ B_F\ \}\ ].$$

By definition,

$\mathbf{N_{sh}} = A_n^1 = N!$

where A_n^p is the number of permutations of p objects among n. N is the number of permutable objects.

Note: in the present work, the dilution is not affected by the symmetries of the objects themselves. These symmetries of the shapes were already taken into account when the variables and their ranges were chosen and thus reduced the variable space size.

The two other factors (N_sol and N_{sym}) are dependent on the CDP and must be studied case by case.

CDP 1 : Squares and Cubes

The number of permutations in these two CDPs (squares and cubes) is equal.

$\mathbf{N_{sh}} = N\_obj!$

Table VII. Number of solutions of the 2D and 3D CDPs in the worst case and the known best cases for N permutable objects.

	worst case		**best case**	
	2D	**3D**	**2D**	**3D**
$\mathbf{N_{sh}}$	N!	N!	N!	N!
$\mathbf{N_{sym}}$	4	8	8	32
N_sol	1	1	?	?
$\mathbf{N\_sol_{CD}}$	4N!	8N!	>8N!	> 32N!

CDP 2 : Pallet Loading

The pallets usually differ in weights, however each time two pallets have the same weight they can be permuted and thus increase the number of solutions. Hence, the worst case happens when no two pallet are alike and the best case happens when all the pallets can be permuted.

Table VIII. Number of solutions of the 2D and 3D CDPs in the worst case and the known best cases.

	worst case		best case	
	2D	**3D**	**2D**	**3D**
N_{sh}	1	1	N!	N!
N_{sym}	1	1	2	2
N_sol	1	1	?	?
N_sol_{CD}	1	1	b N!	b N!

CDP 3 : Circles and Spheres

In this case all the elements are permutable (as in CDP 1 and CDP 1"), however the domain itself cannot be rotated.

Table IX. Number of solutions of the 2D and 3D CDPs in the worst case and the known best cases.

	worst case		best case	
	2D	**3D**	**2D**	**3D**
N_{sh}	N!	N!	N!	N!
N_{sym}	1	1	m^2	m^3
N_sol	1	1	?	?
N_sol_{CDP}	N!	N!	r N! m^2	r N! m^3

CDP 4 : 2D and 3D Pentominos

By definition, none of the Pentominos are permutable however there exists at least 5 solutions for the 2D case and 12 solutions for the 3D case hence respectively for the 2D and 3D cases :

Table X. Number of solutions of the 2D and 3D CDPs in the worst case and the known best cases.

	worst case		best case	
	2D	**3D**	**2D**	**3D**
N_{sh}	1	1	1	1
N_{sym}	8	32	8	32
N_sol	1	1	3	8
N_sol_{CD}	8	32	24	236

CDP 5 : Freeform 2D and 3D cases

The worst case of this CDP appears when all of its elements are different and when each time they move either individually or as a group, the value of the objective function changes. The best case happens when all the components of the system are the same and when the system can move as a rigid body without changing the value of the objective function.

Table XI. Number of solutions of the 2D and 3D CDPs in the worst case and the known best cases.

	worst case		best case	
	2D	**3D**	**2D**	**3D**
N_{sh}	1	1	N!	N!
N_{sym}	1	1	m^2	m^3
N_sol	1	1	?	?
N_sol_{CDP}	1	1	$< N!\, m^2$	$<N!\, m^3$

Summary of the Dilution Analysis

Combined with the variable space size, Dilution lower bounds can be obtained for all the presented CDPs for N and 12 objects (Tables XII, XIII).

Table XII. Dilution for N objects.

		2D		3D
Best Case Dilution	CDP_1	$N! / (1x10^9)^N$	CDP_1"	$N! / (1x10^9)^N$
	CDP_2	$N! / N!(1x10^3)^N$	CDP_2"	$N!/ N!(1x10^5)^N$
	CDP_3	$N!m^2 / (1x10^6)^N$	CDP_3"	$N!m^3/ (1x10^9)^N$
	CDP_4	$24 / (1x10^9)^{12}$	CDP_4"	$236/(1x10^{16})^{12}$
	CDP_5	$N!m^2 / (1x10^9)^N$	CDP_5"	$N!m^3/(1x10^{16})^N$

		2D		3D
Worst Case Dilution	CDP_1	$N! / (1x10^9)^N$	CDP_1"	$N! / (1x10^9)^N$
	CDP_2	$1 / N!(1x10^3)^N$	CDP_2"	$1 / N!(1x10^5)^N$
	CDP_3	$N! / (1x10^6)^N$	CDP_3"	$N! / (1x10^9)^N$
	CDP_4	$8 / (1x10^9)^{12}$	CDP_4"	$32 / (1x10^{16})^{12}$
	CDP_5	$1 / (1x10^9)^N$	CDP_5"	$1/(1x10^{16})^N$

Table XIII. Logarithms of the Best dilution (and worst dilution) for 12 objects CDPs.

		2D		3D
Dilution (Log)	CDP_1	-229 (-229)	CDP_1"	-229 (-229)
	CDP_2	-83 (-102)	CDP_2"	-138 (-158)
	CDP_3	-132 (-145)	CDP_3"	-207 (-228)
	CDP_4	-245 (-246)	CDP_4"	-436 (-438)
	CDP_5	-214 (-248)	CDP_5"	-401 (-442)

The proposed dilution of the solution establishes the difficulty of randomly finding the solution of a CDP. It must be noted that the influence of the objective function is small. The majority of the solutions are coming from the permutations of objects with the same shape and from rigid body motions having no influence on the objective functions. Hence for the ECDPs, the expected number of solutions might be very small and approximately equal to one.

Trend of the Reference CDPs

The objectives bring not only the solutions from which Trend can be deduced but also the Trend. In common CDPs the Trend is clearly identifiable as long as the objects are far from each other's. When the objects are closing each other, the local Trend drops because of the penalty (property 2,p. 69). However, studying the objective function alone is of interest especially when it is combined with the penalty functions. In the CDPs presented so far 2 functions appear the most often : volume objective function (CDP 1 to 5) and center of gravity (CDP 2).

As long as the variables considered are the translations, these two functions have a trend equal to one. However, usually the Trend is unknown. The only proved fact is that Trend drops when penalties are introduced. This change can be evaluated using the dilution of the Feasible Area since this last quantity gives a measure of the portion of the variable space introducing perturbations with respect to the objective function value. Using the property 4 (p. 69) of the Trend, it is possible to link the Trend to the feasible area

$$Tr[F]_{/[Bf]} = Tr[F]_{/fa} + Tr[F]_{/ufa}.$$

$$Tr[F]_{/[Bf]} = Tr[F]_{/[BF]} * Dil_{fa} + (1-Dil_{fa})* Tr[F]_{/[BF]} * k. \qquad (18)$$

where k is a real number between 0 and 1.

In a strategy replacing the objective by the penalty function for each unfeasible point the Trend of the new landscape might be very small (since the first term of equation (18) is close to zero and since, in the second term $Tr[F]_{/[BF]}$ is replaced by the Trend of the penalty function which is null). Such a low Trend compromises the chance to discover a solution with any type of algorithms. This is why it is always better to combine the objective and the constraint in the unfeasible area.

Summary

Except for the conventional surface, volume and center of gravity functions, the objective is unknown and thus no conclusion can be drawn about its Trend. The only information available is the change of Trend (ΔTr) when the penalty function is added. Reducing ΔTr must be one of the objectives of the penalty function design.

The CDPs solution dilution is very small however, these numbers must be compared to the Roughness. Many of the points belong to the same local valley. Moreover, whenever several solutions exist because of the permutation of objects, several of these local valleys lead to a Global Optimum (in the variable space).

Conclusion of this Study

Four factors have been studied to measure the difficulty to solve a generalized CDP. Three of them have been retained for rating this difficulty (Rg, Dil_{fa}, Dil_{sol}) about which the following conclusions are drawn:

- The Dilution of the Solution (Dil_{sol}) is dependent on the number of objects, and on their degrees of freedom. Its bounds are equal to

  $$1/\mathbf{Card}[\ \{ B_F \}\ ] < Dil_{sol} < Dil_{FA}$$

- The Dilution of the Feasible Area links the characteristics of the total System size to the size of the objects. This quantity is the major difference between continuous and the discrete CDPs complexities since the complexity of the latter do not involve this factor. Its bounds are

  $$(-os_{min})^N\ (1/s)\ \Gamma[-s/os_{min}+N] / \Gamma[-s/os_{min}] < Dil_{FA} < (-os_{max})^N\ (1/s)$$

  $$\Gamma[-s/os_{max}+N] / \Gamma[-s/os_{max}]$$

where os_{min} is the size of the smallest component and os_{max} is the size of the largest component of the system.

- The Roughness introduced by the penalty function reflects the number of objects. The total Roughness of a CDP can be bounded by:

$$Rg_{pen} < Rg_{CDP} < Rg_{pen} + Rg_{obj}$$

This factor is the closest equivalence to the discrete complexity measures.

- Rg_{pen} was estimated to grow like an exponential function of the number of interacting objects. In practice, few objective functions reach this level of Roughness, hence the assumption is made that Rg_{CDP} is equivalent to Rg_{pen}.

Unfortunately the Trend is difficult to evaluate even considering the many clues given by the components and system characteristics. The retained criteria were applied to the CDPs chosen as test cases and gave the rating presented in Figure 42.

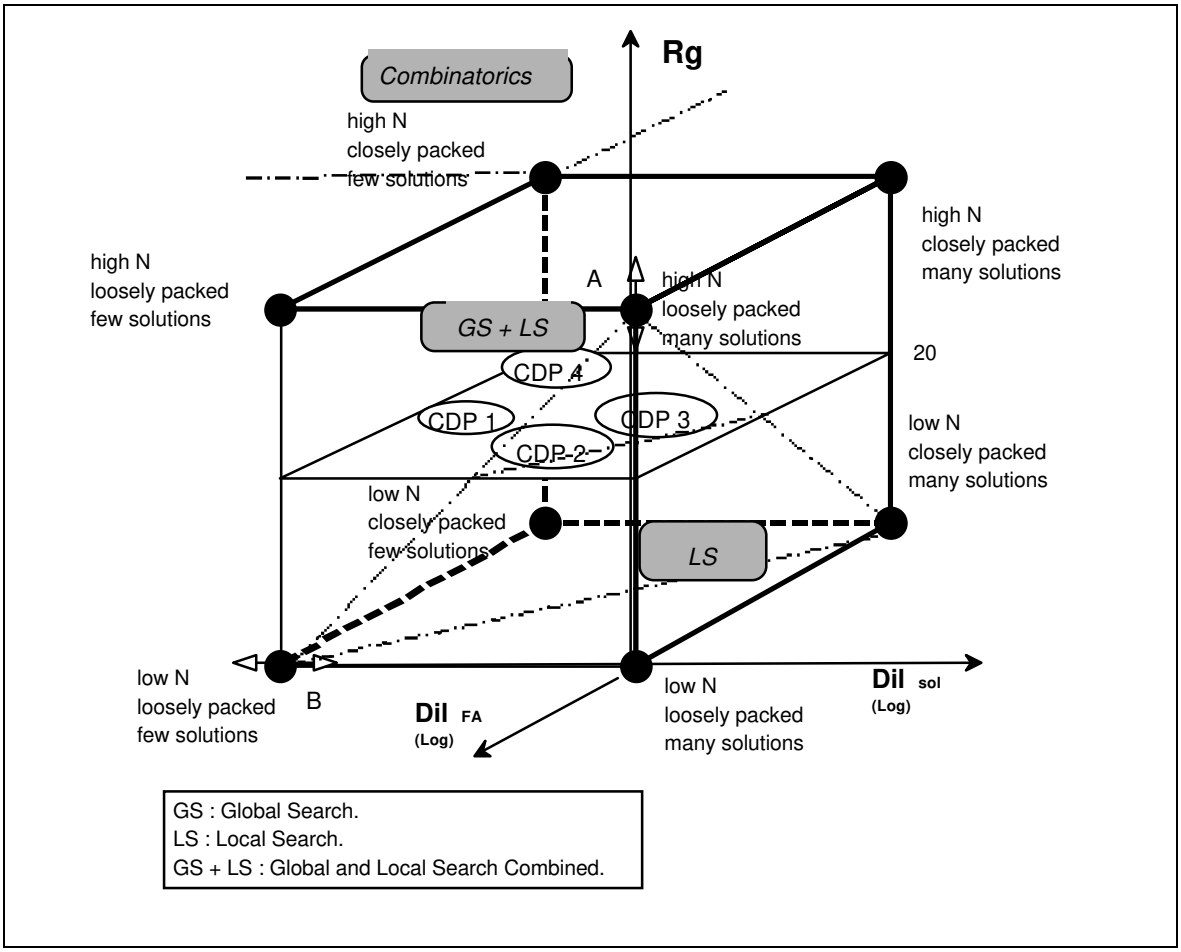

Figure 42. Complexity classification. Three parameters are used to rate and compare the CDPs of the same family. The variation of Dilution of the solution (Dil_{sol}), the Roughness (Rg) and the Dilution of the Feasible Area(Dil_{FA}). When the family (the objective) of a CDP changes the point A and B might move such that the limits defining which method would be more efficient will not be the same.

Depending on the location of a problem on this graph, different fields of mathematics are used to look for a solution. While problems involving few, loosely packed objects could be solved using one of the many numerical algorithms for the optimization of continuous functions (LS), a problem with a high number of tightly packed objects is better solved with combinatorial methods. However, as already stated, these methods are dedicated to discrete CDPs. In between the simple continuous CDPs and the NP-hard discrete CDPs lies a category of problems that still might be addressed by numerical methods involving global and local search (GS + LS).

The Research Hypotheses and Their Verification

Through the validation or non-validation of the 4 working hypotheses, this work will give the answer to the fundamental questions about the optimized placement of parts in mechanical design.

Each hypothesis is formulated as a statement which must be tested. A good hypothesis has two characteristics. First, it must be testable by a procedure and second the answers to the fundamental question it addresses must cover the full field of possible answers. Thus, if the hypothesis is wrong, the range of investigation will have to be narrowed. If it appears that the hypothesis is true, then a new hypothesis must be made in order to refine the scientific knowledge about the subject and so forth until the understanding of the problem is so complete that there is nothing new to do but apply a specific procedure.

In this line of thoughts the following hypotheses are made.

Hypothesis 1: It was shown in the literature review that packing problems have had a long history in discrete optimization and combinatorics. Moreover, Grignon and Fadel (1996) showed that it is more efficient to deal with continuous CDPs as mixed discrete/continuous problems whenever the CDP can be decomposed into several smaller independent problems. However, when dealing with freeform objects, this type of method is inadequate and it seemed that we were brought back to the initial problem of finding the coordinates of the components in a global Cartesian coordinate system with only penalty strategies to prevent intersections between the objects.

The first hypothesis of this work is that, even if the continuous problem formulation and its bigger space size is inescapable for ECDPs, it is possible first to, artificially increase the size of the solution in the variable space such that the chances of success are increased, and second, to reduce the complexity of the landscape - which is mainly due to the penalty functions applied - to drive the search toward feasible regions.

It is proposed to use a combination of Local and Global Search (GS) strategies to test the first part of this hypothesis.

- GS alone with conventional penalty function.
- Coupled GS and Local Search (LS).

From a more theoretical point of view this study will lead to a conclusion about the usefulness of decoupling the way constraints and objective are used for additional reasons than the conventional computational cost of the local search.

It is then conjectured that the use of relative placement of the objects will significantly decrease the complexity of the problem.

The conclusions drawn from the results of this investigation will help to provide an answer to Question 2.

hypothesis 2: We make the assumption that, for a given objective function, the difficulty to solve a CDP is driven by the number of its solutions, the number of the components, their relative size and the total system size according to the classification proposed in the previous paragraph.

The verification of this hypothesis will be done through the use of several test cases involving different number of components, and different number of solutions and different objective functions. The analysis of the results will bring the answer to question 1.

Hypothesis 3: The major obstacle to convergence in the optimization problem comes from the complexity of the landscape (non-linearity of the function, split domains, small feasible area).

At this point of the work, it is useful to recall the work on Complexity of the Nobel Prize Ilya Prigogine. In their book, Prigogine and Nicolis (1989) tell how complex behaviors exhibited by ants colonies searching for food can be explained by the probabilistic behavior of each member of the colony in combination with the configuration of the external landscape. One of their conclusion is that there exists an optimal balance between deterministic strategies and random strategies that leads to the optimal searching performances. This balance is a function of the ants behaviors and of the difficulty to find the food (Nicolis and Prigogine 1989, pp. 232-237). It is interesting to note the parallel between the ants and some modern non-deterministic algorithm for which the best performances are likewise linked to a good balance between the random search and the heuristic search. Considering the optimum of a mathematical function to be found as the food source, and the search method as a colony of ants searching, Prigogine's research shows that the probability to find this optimum is only a function of the size of the optimum and is not related to the non linearity of the function. However, as soon as the search method tries to

make use of some information previously gathered, in order to gain efficiency, then, the complexity of the landscape must be taken into consideration and introduces a bias in the chances to discover the optimum. This bias comes from two sources. First, the information itself which is by definition incomplete since it is a sampling of the landscape, might be so scarce that its analysis will be of no use or worse will point toward wrong directions. Second, the method used to extract the knowledge of the location of the global optimum from this information might be inadequate and mislead the search. It is then not difficult to understand why Genetic Algorithms, Simulated Annealing and other non-determinist search algorithms might behave erratically when problems present large chaotic landscapes.

Similarly to Prigogine's ants, it is assumed in this work that there exists a balance between random search and learning. It is proposed to let the method adapt the range of the search as would a human being do, who, after some time spent for searching unsuccessfully something in an area would enlarge its perimeter of investigation. This study will bring the answer to Question 3.

Hypothesis 4: In engineering it is of interest to get many solutions to a given problem. Many methods providing a large number of solutions were proposed in the literature. For example, a method can be run N times and each time get a different solution by penalizing already found ones. This approach is not really efficient since there is no stopping criteria and since the information reuse from one run to the other is minimum (only bookkeeping of the previous solutions). Interesting solutions were suggested by the existence of population based methods like Genetic Algorithms. Since these methods work on a population of points it was tried to spread the latter at each global optimum of the landscape. However, like a large part of the research in this field is proving, a major problem to overcome is to prevent the trend of GAs to bring points together. That pushed the author to develop a new approach based on population of sets of points instead of single points (Grignon and Fadel 1997). In this work it is assumed that the CDOM implemented with population of sets of points, instead of population of individual points, will overcome this difficulty and that this approach, which has always been applied to MOPs can also be applied to SOPs. This study will bring an answer to the fourth fundamental question.

Conclusion

In this chapter, a single formulation was proposed for solving multi objective constrained optimization problems based on the notion of penalty functions and scalarization to transform the MOP into a single objective optimization problem. Using the implications of this formulation, four criteria namely, Dilution of the Solution, Dilution of the Feasible Area, Roughness and Trend were proposed to quantify the complexity of the new problem. Three of those criteria can actually be calculated on the a-priori knowledge available on the ECDPs (size of the objects, size of the system, number of variables). The fourth criterion (Trend) requires, to be evaluated, the knowledge of the objective function at all points of the domain which is impossible otherwise the ECDP would be already solved. However, an approximate value of the Trend can be calculated using the CDPs characteristics (size of the system and number and size of the system components). Hence, a study of these criteria was performed in order to identify which CDPs might be easier to solve than others and for what reason. The study of these measures on well-known CDPs helps to identify several research hypotheses on the strategies that might decrease the ECDPs complexity i.e. Relative Placement for the variables, and Local Search for the method. In the next chapter, the method CDOM is described in detail. The scalarization of the multi objective ECDP is explicitly given as well as the Genetic Algorithm modifications and parameters. Several method variations, corresponding to the hypotheses, are explored. The study of the complexity criteria is used to provide a formal basis to the verification of the hypotheses.

CHAPTER IV

METHOD DESCRIPTION

The Configuration Design Optimization Method

The CDOM implements several techniques in order to tackle the main difficulties identified during the CDPs analysis. Among those techniques, a proposition is made for the definition of new variables, as well as different search strategies combining local and global search. The method is also designed to provide multiple solutions spread along the Pareto set.

It must be emphasized that the construction of an optimization method that would guarantee the exact solution to intractable problems such as CDPs is presumptuous. CDOM must be put in the category of the approximation algorithms for which there is only a probability to reach a satisfactory solution. This method can also be viewed as a manager that has a finite amount of resources to reach a goal. The algorithm must manage its own resources, which are storage capacity and time, in order to extract and exploit enough information to reach the solution as fast as it can.

Method Description

Populations of Sets and Genetic Algorithms

The Configuration Design Optimization Method (CDOM) was first designed to cope with the non-linearity of packing problems with two objective functions (Center of Gravity and volume, Grignon, Wodziak and Fadel 1996). The Genetic Algorithm, combined with a mixed discrete/continuous model, was already the tool of choice in this domain and was competing with rule-based methods. More over, its ability to deal with different types of variables, which is common for CDPs, was confirming this choice for configuration problems.

When considering different objective functions and the willingness to add more than two objectives, it was clear that the GA, which allows the gathering of several solutions in a single run, had another advantage with respect to other global optimization techniques. However, the different techniques proposed in the literature were not sufficient to cope with a continuous set of solutions like a Pareto Set.

The major problem encountered by researchers trying to use GAs to obtain multiple solutions at the same time is the tendency of GAs to bring points together. In order to cope with this problem several strategies were proposed by the Genetic Algorithm community ranging from using sharing in the objective space to the selection of sub-populations of points proportionally to each objective in turn (Schaffer, 1985) or to the definition of a new objective function based on the definition of the Pareto set (Fonseca et al. 1993). Whatever the method proposed, the competition between the initial design of the GA, i.e. gathering points around the global optimum, and the method to spread the points of the population was still occurring with various outcomes none being completely satisfactory. A more complete literature review on this subject is available in Grignon and Fadel (1998 - submitted to journal).

Hence, instead of using a population of points (Figure 43) it was decided to remove the difficulty of obliging points to stay away from each others by using populations of sets of points.

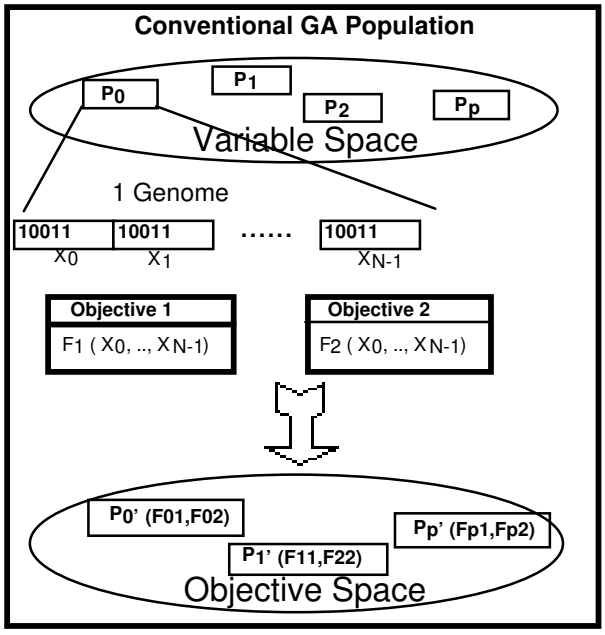

Figure 43. Conventional GA population. During one generation, the conventional GA attributes a grade for each point of a population.

In this approach (Figure 44), each member of the population is no longer a single configuration but a set of configurations. The advantage of such a choice resides in the fact that the valuable solution, from the GA point of view, is no longer a single point but the shape and position of a cloud of point. The best solutions are the clouds well spread along the Pareto Set.

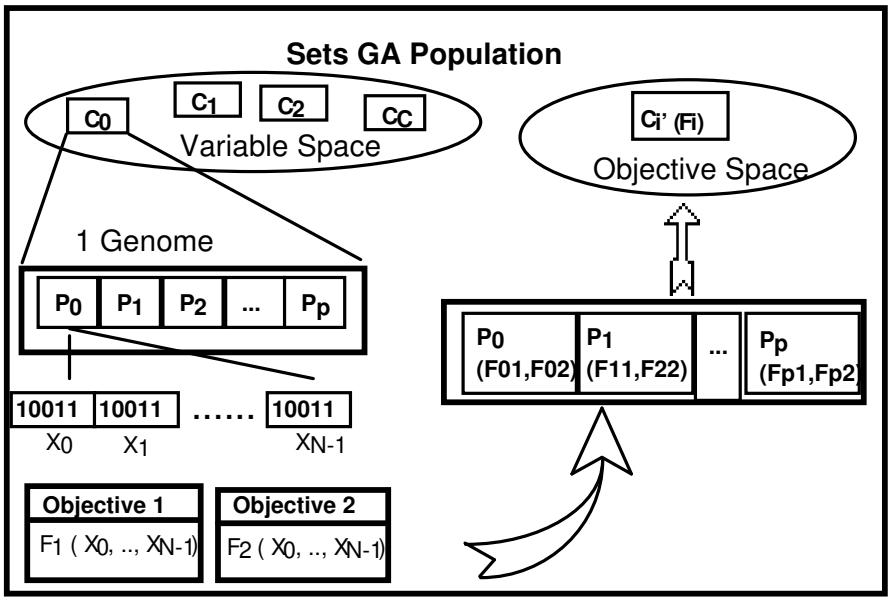

Figure 44. Sets GA populations. The Sets GA attributes a grade for each cloud based on the relative position of each point in a cloud. This is done for all the clouds constituting a population.

Such formulation brings different problems. By making the size of the genome bigger, the limitations of the GA were reached quicker and the rate of convergence was degraded (Grignon and Fadel 1997). In order to overcome these new problems of convergence and to exploit the fact that the GA had very good performances only during the first few hundreds generations and then levels off, the algorithm was made iterative (Figure 45).

At first glance it seemed that this choice was only making the method slower for getting back a decent rate of convergence and acceptable results. At a second glance the iterations were opening the search process to an eventual interaction with the user. Thus, the

experience of engineers, which is a level of knowledge that an automated method has few chances to reach, is now available for the CDOM. This was made possible by changing the bounds of each variable after each run of the GA in order to determine the new domain of interest for the next run. These new bounds are determined by the extreme points of the best approximation of the Pareto set discovered so far by the method and define the variable ranges for the next iteration. These ranges are increased by 20% in order to allow the CDOM to expand the Pareto set. The final argument for using a GA was its versatility to switch from pure random search to pure learning search which allows the addressing of a broader category of CDPs.

Hence, the optimization kernel of the CDOM is a GA in which each individual is a set of points taken in the domain of the CDP ($[B_F]$).

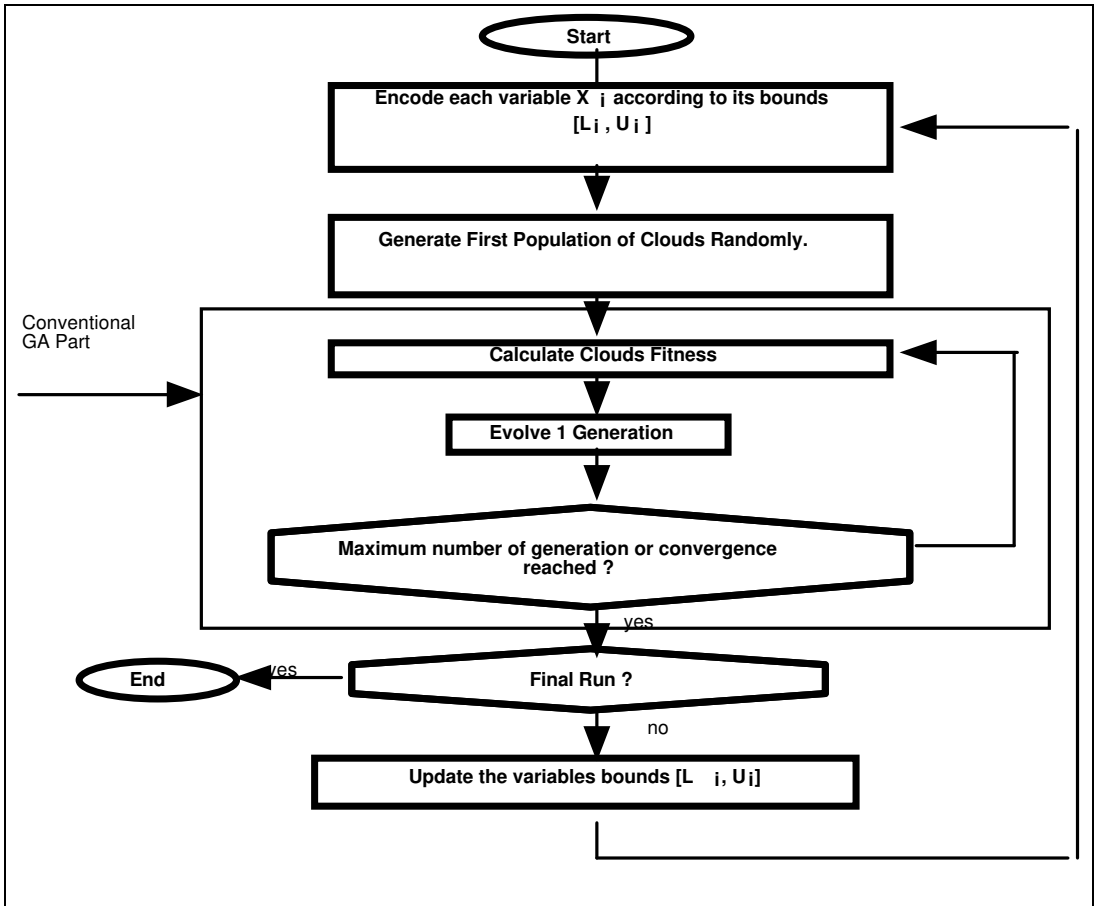

Figure 45. CDOM flow chart.

Variables

All the variables involved in the CDPs presented so far were the positions of the components with respect to an absolute Cartesian coordinate system. In order to respect the functional constraints involved in many mechanical CDPs it was decided to replace the absolute positions variables (x, y, z, rx, ry, rz) by a translation vector **T**(tx, ty, tz) and a rotation vector **R**(rx, ry, rz) with respect to a coordinate system attached to another component. Both combined give a relative displacement tensor, {**T**,**R**} easy to represent (Figure 46) and which allows to position exactly a component in space.

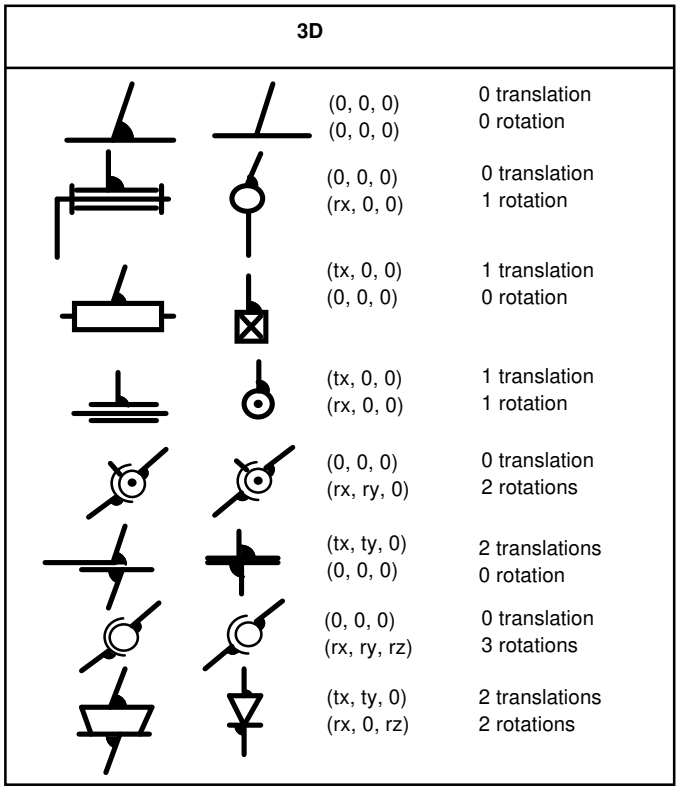

Figure 46. Correspondence between the Mechanical Functional Links and the Displacement Vectors.

This way of placing components offers the same possibilities as the absolute coordinates (Figure 47, 48), it also gives the possibility to implicitly satisfy many

mechanical functional links that, otherwise, would have been satisfied by equation solving (Figure 49). However, the choice of the movements attributed to a component with respect to another one is left to the engineer. Thus, he or she is in charge of defining displacements that do not contradict the Mechanical Functional Constraints. In addition to the first advantage, this choice of variables is closer to a mechanical sense of what it is possible to ask for the displacements of a component.

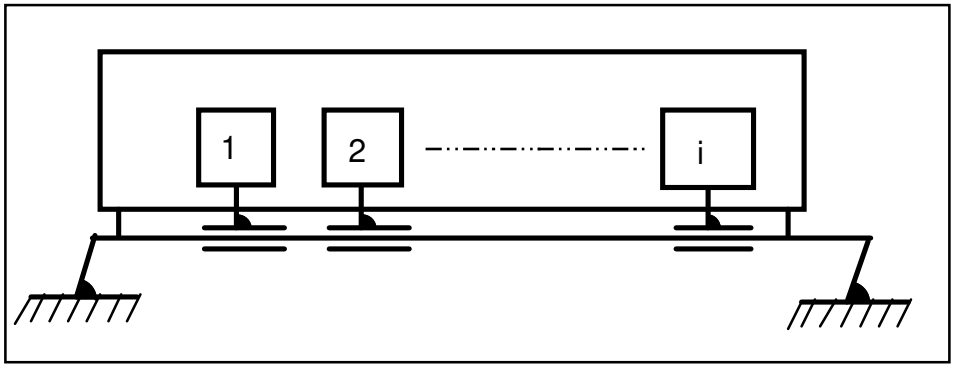

Figure 47. The one dimension Pallet Loading CDP (CDP 2) with relative positioning vectors. In this case the relative and absolute positioning are the same.

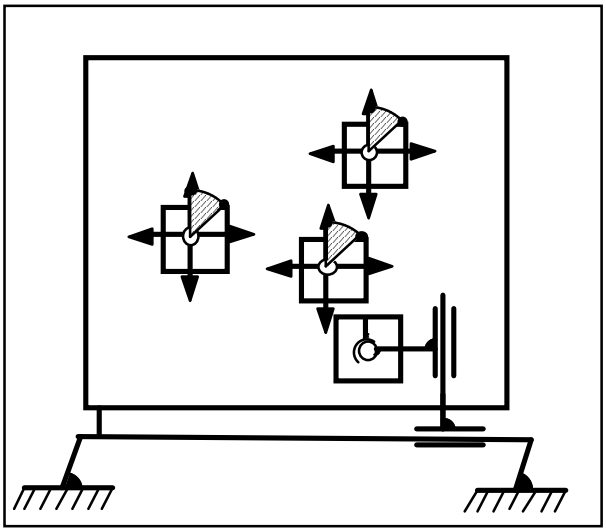

Figure 48. The two dimension Square CDP (CDP 1) with absolute positioning vectors.

In the car engine example (Figure 49), the problem consists of placing the main components of the car underhood (motor, radiator, fan, fan engine, battery, power steering pump and tank, alternator, air-filter, and master cylinder) while trying to optimize the compacity, the position of the center of gravity and the maintainability of the system. Addressing the simplified 2D version of this CDP as an absolute components placement problem requires the use at least 11 non-linear equality constraints involving 18 translations.

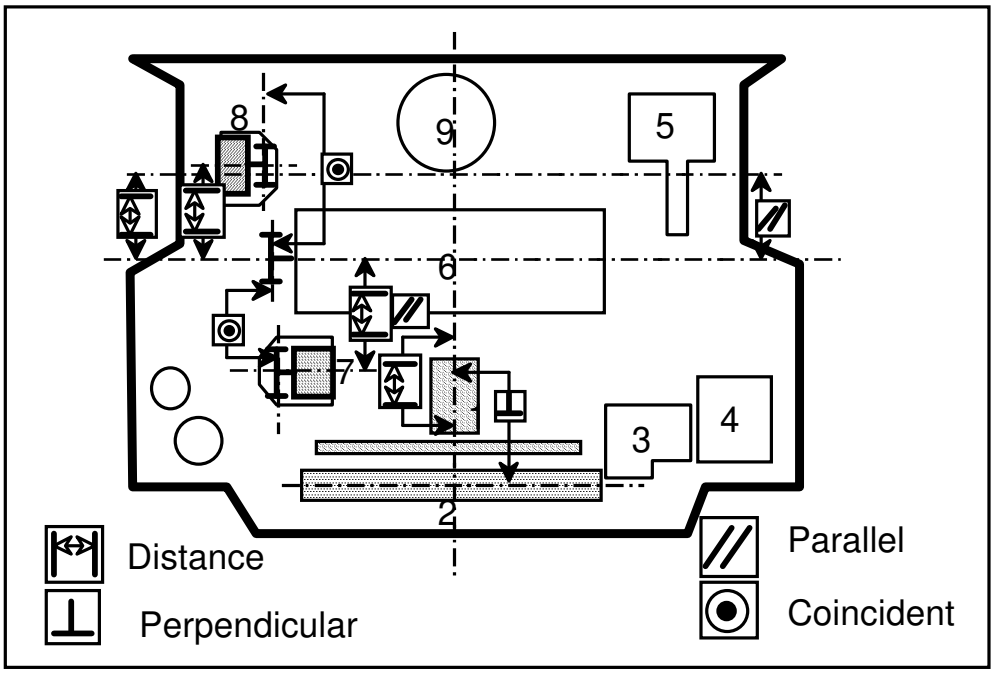

Figure 49. The 2D car engine problem. The components are placed using a global coordinate system and geometric constraints.

According to the equations proposed in this chapter the Roughness criterion for such a problem is:

$$Rg \cong 11.34$$

Addressing the same problem with relative placement reduces the number of variables to 13 (Figure 50), automatically satisfies the Mechanical Functional Constraints and reduces the Roughness to 7.6.

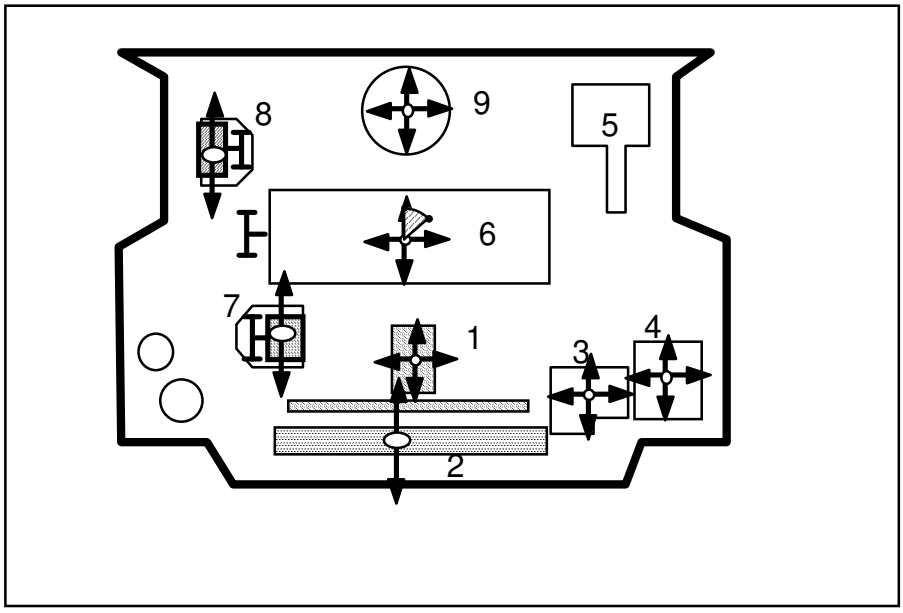

Figure 50. The 2D car engine problem with relative placement. The components can move relatively to each other's.

Influence of the Relative Placement on Roughness

The analysis of this expression shows that first the number of interactions between objects can be reduced by using relative placement instead of absolute placement. This is due to the fact that, when relative placement is used for one component instead of absolute placement it is assumed that it cannot overlap with its positionner. However, this gain is small (Figure 51) since the number of intersections of absolute and relatively placed objects both grow like a square of the number of objects of each type.

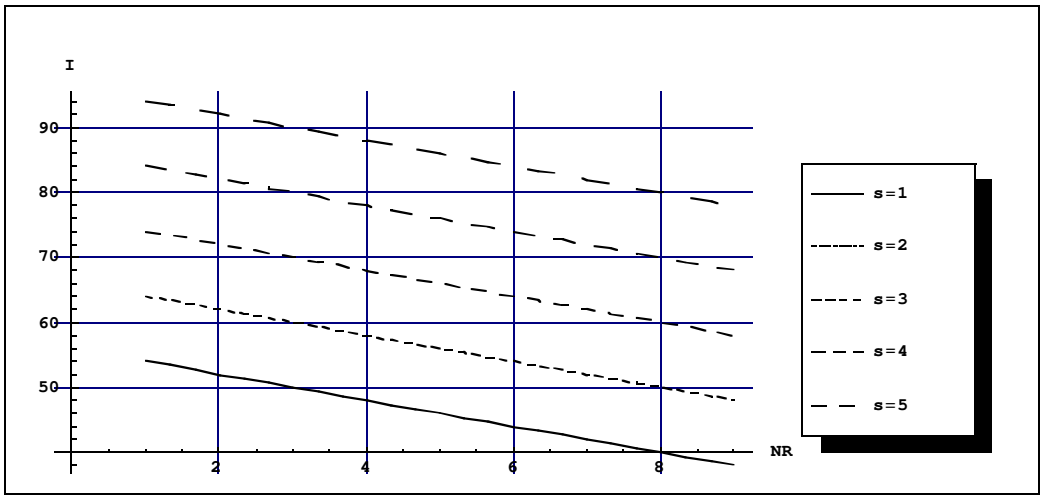

Figure 51. Evolution of the number of interference. This number (I) decreases when component positions are translated from absolute positioning to relative positioning (NR). The letter 's' is the number of static objects.

The second interest in choosing relative placement instead of absolute one comes from the fact the positioning of mechanical objects can then respect mechanical functional constraints. These constraints ensure that the overlapping does not occur between a component and the components placed with respect to the same positionner (see next paragraph for a definition of Mechanical Functional Constraints). Hence equations (10) and (11) must then be reviewed since the components are not interacting with all the others in the following way.

Let's call A a set of 'a' components with an absolute positioning system, R a set of 'r' components with a relative placement, and S a set of 's' static components. Each of the members of A can now interact with a subset A' of 'pa' components of A. They can also interact with a subset R1' of 'pr_1' components of R and with a subset S' of 'ps_1' components of S. Then, the components of R can interact with a subset R2' of 'pr_2' components of R and with a subset S2' of 'ps_2' components of S. This changes the number of interactions in the following way:

$$N_{int} = (pa-1)(pa-2)/2 + (a - pa)pa + (a - 1)pr_1 + a*ps_1 + r*pr_2 + (pr_2-1)(pr_2-2)/2 + r*ps_2. \quad (19)$$

$$N_{int} = (pa-1)(pa-2)/2 + (a-pa)pa + (pa - 1)pr_1 + a*ps_1 + r*pr_2 + (pr_2-1)(pr_2-2)/2$$

$+r^* ps_2$.

The gain thus obtained is important for huge mechanical assemblies in which sub-systems keep their components inside a limited area. Sub-systems have few chances to interact with many others and so the interaction of their components is usually limited to the components of their immediate neighbors which means that pa is small in front of a, pr_1 is small compared to r, pr_2 is small compared to r. This lead to a the following simplification of equation (19)

$$N_{int} = a^*pa + a^*ps_1 + a^*pr_1 + r^*pr_2 + r^*ps_2. \qquad (20)$$

This equation can be compared term by term with significant terms of equations (14) and (15) which gives the following correspondence:

Table XIV. Comparison of significant terms of equations (14) and (20).

from Equation 10	from Equation 20
a^2	a*pa
a*s	a^*ps_1
a*r	a^*pr_1
r*s	r^*ps_2
r^2	r^*pr_2

The terms pa, ps_1, pr_1, ps2 and pr_2 are usually fractions of the initial number of components and their value levels off for huge mechanical systems. In the worst case pa = pa_{max} = a, ps_1 = s, ps_2 = s, pr_1 = r, pr_2 = r. However, depending on the system considered these parameters can be associated with an average value which allows to estimate an average number of intersecting components. Finally the best case occur when pa = 1, ps_1 = 1, ps_2 = 1, pr_1 = 1, pr_2 = 1 which gives a number of interactions equal to 3a -2 + r.

This expression shows that placing objects relatively to each other's instead of leaving them free to move all over the domain does not bring a significant advantage as long as the worse case is considered. However, for an average case there is tremendous interest in reducing the number of interactions between static objects and dynamic ones (a + r). This occurs when ps_1 and ps_2 are small.

Influence of the Relative Placement on the Dilution of the Feasible Area

In addition to reducing the Roughness, the variable space size, choosing relative placement has the additional advantage of increasing the dilution of the feasible area. This is the consequence of the lower chance of getting an interference and of the reduced ranges of the variables.

Objectives

In order to fit the need of the GA for a single objective function value attributed to each point, a scalarization based on two components is proposed. First, a value is calculated in order to differentiate Pareto sets from other sets of points. This scalarization attributes better values to relative non-inferior points. Second, this scalar value is modified in order to take into account the spreading of the points.

The morphology is measured by giving a rank based on the notion of non-inferiority. Each solution point is considered in the objective space and has a rank calculated by comparing its position with the position of its cloud mates and with a reference Pareto set. The reference Pareto set is made of all the best points (non-inferior) found so far (Figure 52). Since the number of points in this reference set is bounded (because of the limited memory size) but not fixed, the grade of the cloud is normalized with respect to the number of points. Thus, for a cloud of N points and a reference set of M points, the grade of the cloud is

$$Grade_{cloud} = \sum_{i=1..N} [\ rank(P_i)\ ] / (\ N + M\ )^2 \qquad (21)$$

Rank[P_i] is N*M minus the number of points dominating P_i.

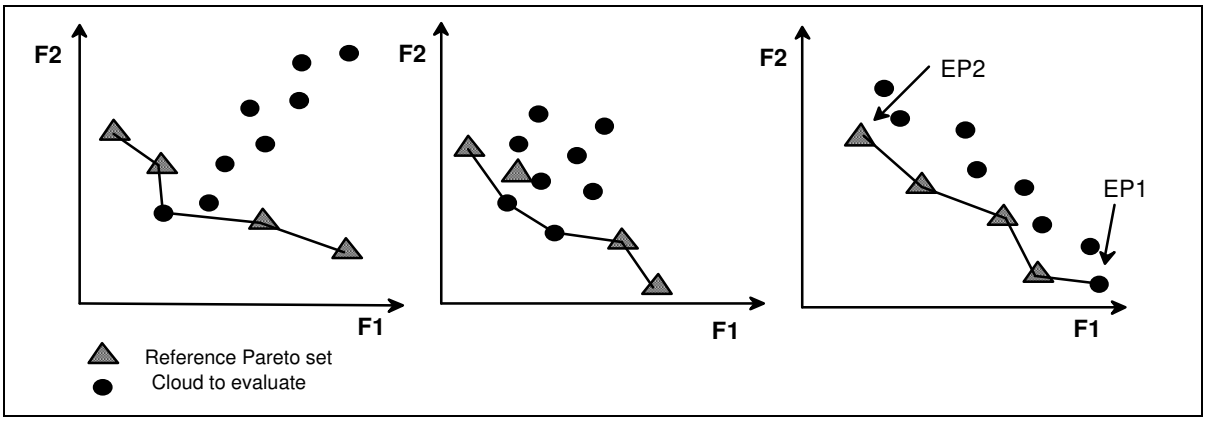

Figure 52. In this example the goal is to minimize F1 and F2. At a given moment the reference Pareto set can be made of points belonging to the previous step reference set and to the cloud being evaluated. From left to right, M = 5, 5, 5, N = 8 ,8, 8 and the clouds grades are 98/169 = 0.58, 132/169 = 0.78, 136/169 = 0.80.

Hence, so far, the grade of a cloud is an integer number. The spreading was introduced by a penalty that always gives priority to the morphology of the cloud since its value is chosen such that it remains between 0 and 1.

$$penalty_{sp} = Atan\ [\ Max[1/Distance[P_i, P_j]\ ]\ ] / (\Pi/ 2)\ i=1..N,\ j=1..N. \qquad (22)$$

The cloud grade then becomes equal to

$$Grade_{cloud} = (\ \sum_{i=1..N}\ [\ \mathbf{rank}[P_i]\ ]\ ) / (\ N + M\ )^2 + penalty_{sp}. \qquad (23)$$

A special interest must be put into the recording of the extreme points of the reference set (Eps, Figure 52c). In order to get solutions as close as possible from the bounds of the Pareto set, the EPs are only replaced by better extreme points such that the spreading of the Reference Pareto Set is always the best spreading found so far. This strategy guarantees a convergence toward the real extreme points (TEPts) which are unknown and which correspond to the solutions of the single objective CDPs involving each objective in turn.

Constraints

It was decided previously in this chapter to translate the constraints into penalties. These Penalty functions are usually added to the objective function as soon as the unfeasible

area is reached (exterior penalty function, Figure 48). It can also be added before the unfeasible area is reached and thus favors the discovery of feasible points (interior penalty function) (Haftka and Gurdal 1993, pp. 186-201). A first order or second order optimization method is then used to quickly retrieve the feasible domain. However this approach can be used only if the calculation cost of the objective function is moderate since first and second order numerical minimization methods require a high number of functions evaluations. A second impediment for the use of this method comes from the fact that the Engineering CDPs are multicriteria problems. Applying the Local Search to each objective in turn leads to three different points in the variable space, and applying the Local Search to the Ranking objective function is impossible since it is discrete.

Replacing the objective function by the penalty alone in the unfeasible region seems to solve all these drawbacks (Figure 48). However, this decreases the Trend of the problem since it removes points bringing useful clues with points bringing either no clues or bad clues (property 2, p. 69). Since the Dilution of the feasible area might still be small, a global method might exhibit premature convergence far away from the useful solutions (Chapter III, pp. 95-99).

Among these two options for exiting the unfeasible area, the choice depends on the use of the Local Search. The penalty alone must be considered in a strategy using a Local Search while the penalty combined with the objectives can be used by a Global Search. In both cases, the value of the penalty is applied to all the objectives at the same time in order to guarantee that no unfeasible point will ever be inside the Pareto set. The Grade of the cloud is still given by equation (23) but the points Pi are replaced by points Pi' such that

$$Pi'[\, F1 + penalty_{int}, \ldots, Fn + penalty_{int} \,] \quad (24)$$

where n is the number of objective functions.

Richardson et al. (1989) recommended the use of a penalty function proportional to the degree of violation rather than proportional to the number of constraints violated. Hence, the interference penalty function is defined by the amount of overlapping rather than the number of objects overlapping. In any case, the penalty is added to a constant factor that makes it worse than the worse feasible configuration (Figure 53 upper left) or it is combined

with the objective and a constant factor (Figure 53 right). In this example a pallet can move along the x-axis while 4 pallets are fixed. The goal is to place the center of gravity of the load as close as possible from the position x=5.5. At left, the landscape of this CDP is composed of either the penalty alone or the objective function alone. At right, the penalty is combined with the objective. The aim of the interference penalty function is to attribute worse values to the configurations with overlap than the worse configuration without overlap.

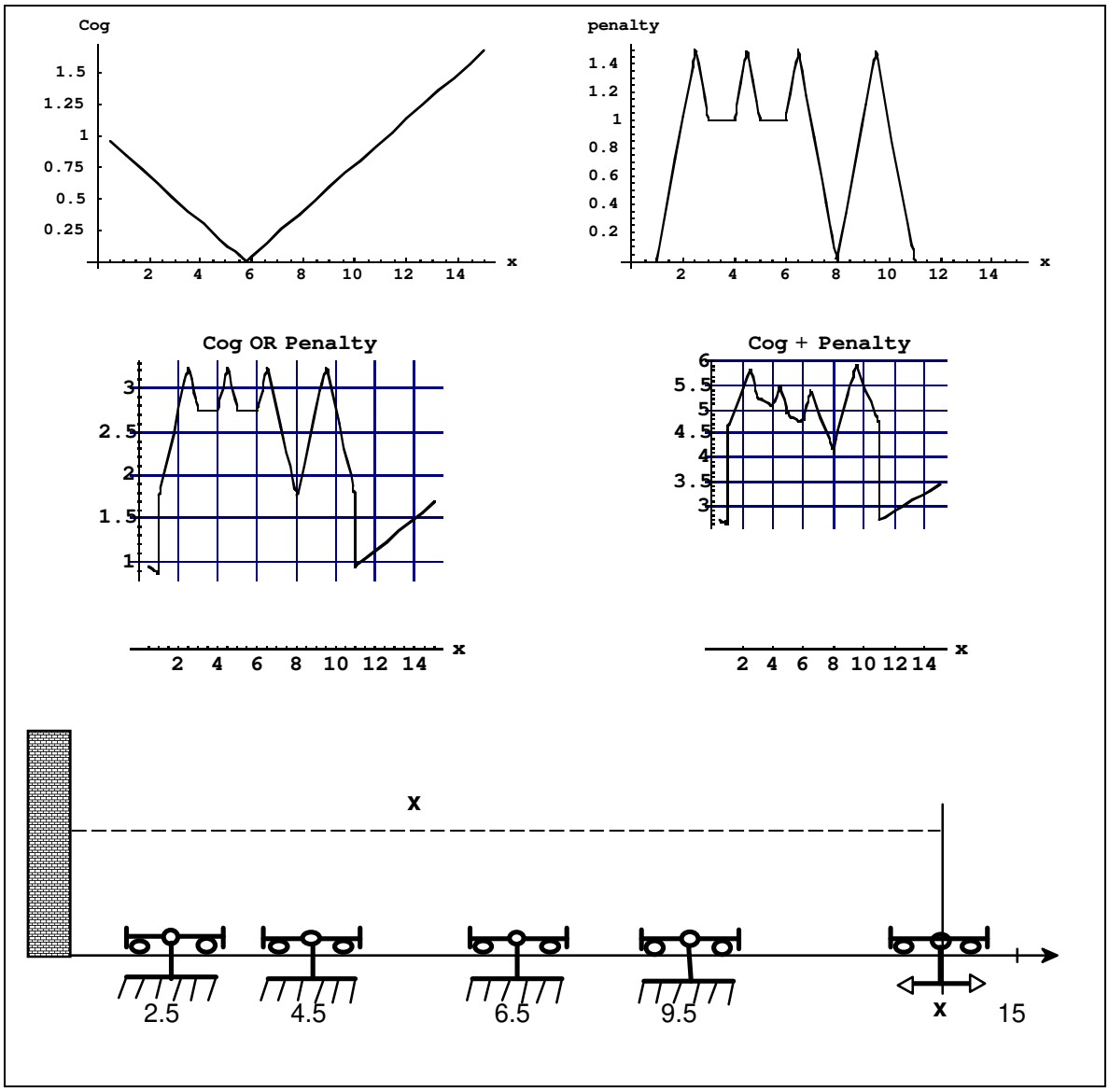

Figure 53. The 1D rolling pallet example

The Different Search Strategies with GA

The experience in CDPs proves that in many cases, the size of the unfeasible area is larger than the size of the feasible one. This is especially true for volume and compactness optimization problems that tend to bring the objects together. For these cases the knowledge of the properties of the unfeasible region helps to design more efficient algorithms. Let's examine how several alternate methods, combination of Local Search and Global Search, can be applied to an initial landscape in order to reduce the size of the unfeasible area and to increase the size of the solution (Figure 54).

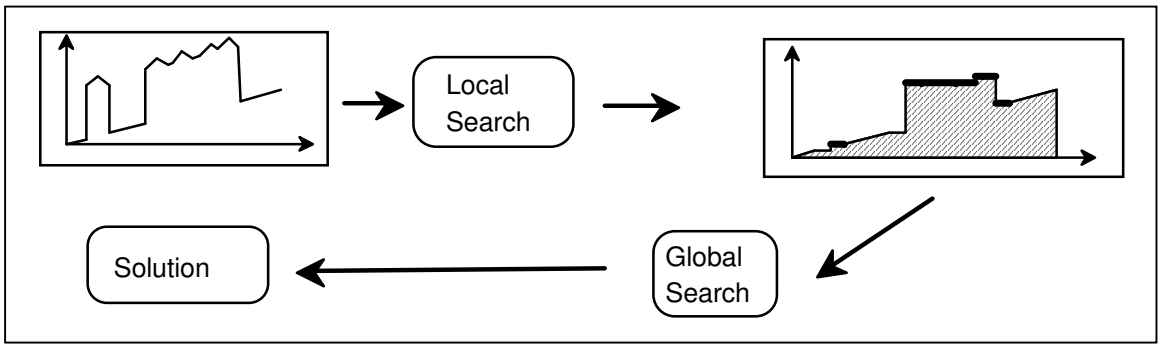

Figure 54. Landscape generation. From the initial problem landscape, a Local Search method is applied to obtain an intermediate landscape on which the Global Search is applied.

Common strategies combining Global and Local consist of applying the Local Search to all the points investigated by the global algorithm. The global optima become as big as their local valleys since the value of the global optimum is assigned to all the points belonging to its valley. This strategy, in addition to the fact that it cannot be applied to a multicriteria problem, has the additional drawback of leveling off the feasible and unfeasible area such that comparing two points in the same valley does not bring any information about a possible evolution (Figure 55).

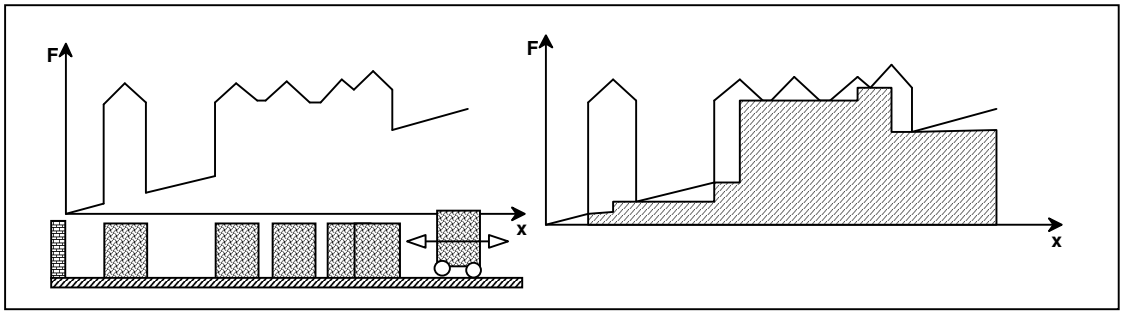

Figure 55. A simple landscape generated by a one Pallet problem. The initial function landscape at left presents feasible and unfeasible regions. The goal of the problem is to find a minimum. At right the intermediate landscape (dashed region) is represented generated by the application of the Local Search to all the points of the variable space.

The strategy which is preferred in this work consists of applying the local search to the unfeasible area only in order to find the closest feasible point. This method has the double advantage of increasing the feasible area (indeed, all the unfeasible points that are in the same local valley as a feasible point become feasible since the local optimum is the feasible point) and of increasing the size of the solution (Figure 56). The computational burden of the local search is reduced since it is not applied to the objective function.

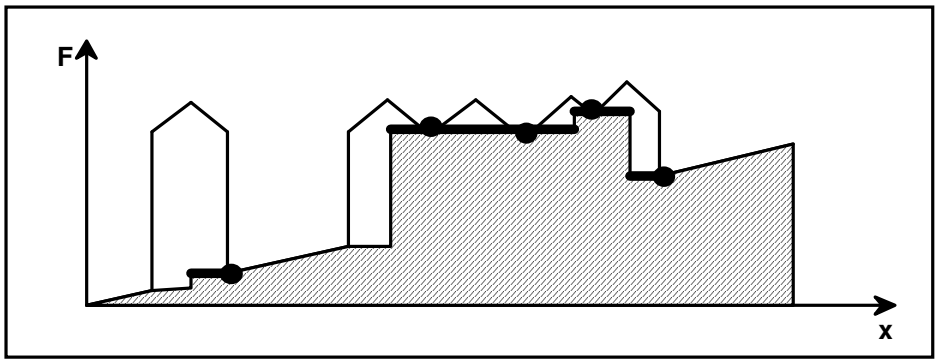

Figure 56. Feasible area size and local search. Applying a Local Search to the penalty function alone increases the size of the feasible area and only levels off the unfeasible area.

In its implementation this method is not straightforward and requires additional choices that are detailed in the next section.

A third strategy is based on the observation that the local valleys of the unfeasible regions that have a size (δ) smaller than the smallest dimension of the smallest object do not lead to global optima. Hence, using the famous Shannon theorem (1948), it is not necessary to discretize further than δ/2 since these valleys are not of interest for finding the solution of the CDP (Figure 57). An advantage of this method is that the penalty remains combined to the objective function in the unfeasible region which results in a higher Trend (property 2 p. 69). A global search can then be applied alone (Figure 58).

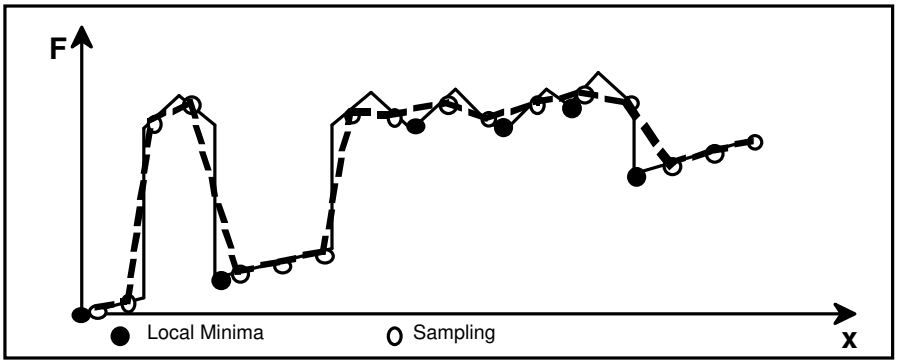

Figure 57. Landscape and discretization. It is not necessary to discretize further than δ/2 in order to keep the interesting features of the landscape i.e. the feasible areas.

A first drawback of this solution is that the best point found is at least at a distance δ/2 of the global optimum if nothing else is done.

A second drawback comes from the fact that the chance to get a deceptive problem increases (property 2 p. 69, Appendix A).

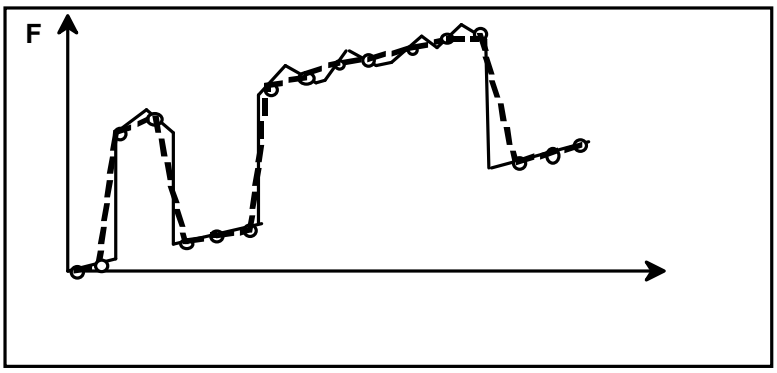

Figure 58. Objective with penalty function and Trend. Adding the objective function to the penalty function allows the introduction of a bigger trend factor for the unfeasible area.

Implementation

Genetic Algorithm

For the reasons mentioned earlier, a Genetic Algorithm was chosen as optimization method. Galib-2.4 is a C++ library developed at the Massachusetts Institute of Technology by Matthew B. Wall (1996). It offers three different evolution schemes -steady state (population overlapping), simple (no population overlapping, replacement (population overlapping with customized replacement)- and four different genomes -binary strings, arrays, trees and lists- as well as different scaling schemes and selection processes.

The GA used in the CDOM is a Steady State GA i.e. a GA with overlapping populations. The overlap of each population is 25%. The selection scheme is a Roulette Wheel selector which selects individuals proportionally to the value of their fitness (Goldberg 1989). The scaling method consistently used along all this work is Sigma Truncation and the crossover is a single point crossover. The population size is of 15 members and the number of designs per child is 15. The genes length is 10 bits. This choice provides a precision of $L/2^{10}$ (roughly L/1000) on each variable, where L is the size of the variable interval of variation during the current run.

Due to the fact that each genome represents in fact a set of 15 different configurations (and each configuration being itself defined by N variables), the total length of each individual genome is equal to 10*(N)*15 bits. The stopping criterion is based on

convergence. A failure to improve the best fitness of 0.001% stops the current GA run and possibly starts a new one.

Local Search Optimization Method: Powell's Direction Set

A zero order method was preferred to first and second order techniques since first the computational cost of a Hessian matrix is prohibitive and second since the differentiability of the penalty functions could not be guaranteed over the whole variable space. This method can be summarized as follows:

1. Given a starting point $\mathbf{P_0}$ and an initial direction $\mathbf{V_0}$ for the search.
2. Find λ such that $f(\mathbf{P_0} + \lambda \mathbf{V_0})$ is minimum.
3. Start again with $\mathbf{P_1} = \mathbf{P_0} + \lambda \mathbf{V_0}$ and $\mathbf{V_1} = \lambda \mathbf{V_0}$.

Step two is done through the use of a Parabolic Interpolation (Brent 1973).

The goal behind using a local search is to identify the closest feasible point. However, the landscape of the unfeasible area presents two characteristics that prevent the success of the search. First, it is non-linear, and second large flat areas replace the feasible domain. The first characteristic traps the local search inside the unfeasible domain while the second traps it into the feasible area. This second state should not be a problem if it were close to the starting point however nothing guarantees that it will be.

Hence, the local search must be used with care. While there is no straightforward way to prevent convergence inside the unfeasible area, it is possible to update the landscape searched by the local search such that it will converge toward a boundary point between feasible and unfeasible domain. Since the penalty translates the fact that two objects intersect, the best way to keep them close-by with no-interference, is to apply the local search only to the interfering objects and to a combination penalty/volume objective function in which the volume is calculated with the involved objects and replaces the penalty when the local search investigates feasible points.

Objects Shapes

In spite of the nearly 20 years of existence of CAD software, it is since very recently that the objects used in configuration optimization upgraded from simple shapes like cubes, blocks, spheres, circles, and squares to 'free-form objects'.

Lomangino in (Lomangino 1994) and Szykman and Cagan in (Szykman and Cagan 1994b), introduced the use of CAD to represent the more realistic objects necessary in engineering. Even if more cumbersome, this approach allows the engineers to take advantage of the piecewise polynomial or rational models implemented in CAD to build free-form geometric objects. CAD are able to generate several representations of the objects in order to save, retrieve and exchange data with other software. Among the standard formats used to exchange information about the shape of an object, one of the more common and easy to use is the tessellated description of the surface of the object (STL format). This type of description is used for the rapid prototyping models, the finite element models and for the visualization models (used in CAD for example). Hence, in this work, each object is described by its skin paved with the triangles obtained by exporting an STL file from a CAD software (Figure 59, 60).

```
facet normal 1.000000 -0.000000 -0.000000
    outer loop
      vertex 23.6220 39.3700 -39.370079
      vertex 23.6220 55.11811 -39.370079
      vertex 23.6220 55.1181 -23.622047
    endloop
  endfacet
  facet normal -1.0000 -0.0000 -0.000000
    outer loop
      vertex -23.6220 55.1181 -23.622047
      vertex -23.6220 55.1181 -39.370079
      vertex -23.6220 39.3700 -39.370079
    endloop
  endfacet
  facet normal 0.000000 1.0000 -0.000000
    outer loop
      vertex 23.6220 55.1181 -23.622047
      vertex 23.6220 55.1181 -39.370079
      vertex -23.6220 55.1181 -39.370079
    endloop
 endfacet
```

Figure 59. The STL file format.

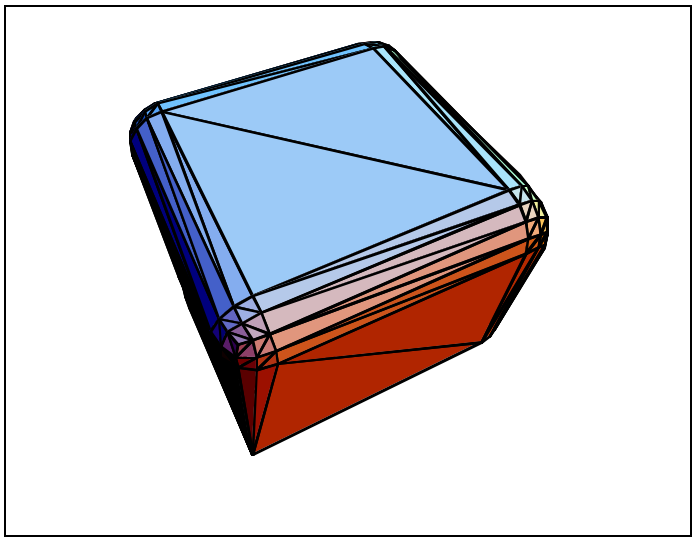

Figure 60. Example of an object used by the CDOM. A tessellated surface of a cube with filleted edges made of 262 triangles.

The advantage of this triangulated representation is that it provides a good balance between the accuracy of a solid description and the computation speed of various functions involving the objects like volume, interference, and clearance. Many fast optimal algorithm of computational geometry are based on this representation (O'Rourke 1993).

The use of more precise representations like parametric surfaces, is impossible due to the calculation time required to compute the interference.

Configuration Optimization Problem Objectives

A mechanical system possesses static characteristics like surface, volume, mass, inertia and maintainability. These characteristics change only when the position of the components changes. Their mathematical representation relies either on analytic expressions or on the numerical integration over the meshed solids.

Length, Surface and Volume Models

The first mathematical model used to measure the volume occupied by one or several objects is the volume of the bounding box. A bounding box is a box in which the objects can be enclosed. If one imagines the objects described in a 3D Euclidean space with a Cartesian coordinate system, then the volume of a bounding box is

$$V_{bb} = (\mathrm{Max}[x_i] - \mathrm{Min}[x_i]) \; (\mathrm{Max}[y_i] - \mathrm{Min}[y_i]) \; (\mathrm{Max}[z_i] - \mathrm{Min}[z_i]). \quad (25)$$

in which x_i, $y_{i,}$, and z_i, are the coordinates of an average point of the surface of any object.

This objective function is used for CDP_1 and CDP_4. However, this definition has several drawbacks. First, it varies when the axis of the coordinate system changes and, second, it leads to results that are far away from the real answer.

A second way to evaluate the volume of a set of objects is to measure the volume of its convex hull i.e. a kind of rubber bag one can wrap around the objects. The volume measured is invariant when the coordinate system changes however it is much more difficult to evaluate. Using this kind of objective to minimize the volume occupied by a set of objects can lead to unexpected, however valid results, which have nothing to do with an admissible mechanical solution (Figure 21). This points out that the notion of compactness in mechanics is a different notion from the notion of minimal volume. Wrongly, this notion has been associated with the ratio of the sum of the objects volumes (or area) with respect to the bounding box volume (or area).

Hence, if the objective is not to reduce the volume but the compactness of the system, the inertia matrix must be used instead of the volume. Its mathematical definition is given by:

$$[I] = \begin{pmatrix} \int (y^2 + z^2)\,dm & -\int (xy)\,dm & -\int (xz)\,dm \\ -\int (xy)\,dm & \int (x^2 + z^2)\,dm & -\int (yz)\,dm \\ -\int (xz)\,dm & -\int (yz)\,dm & \int (x^2 + y^2)\,dm \end{pmatrix} \quad (26)$$

where the integration is done over a solid object.

The matrix is computed for each object and the whole system inertia matrix, which is the goal of this objective, can be calculated by adding the individual matrices as long as they are expressed with respect to the same coordinate system. Thus,

$$[\,I\,]_{system\,/\,ref0} = \sum_{i=1..N} [\,I\,]_{components\,i\,/\,ref\,0} \quad (27)$$

When a transformation $[\,M\,]_{/\,ref\,i}$ is applied to an object , its new inertia matrix is calculated by the following way

$$[\,I\,]_{components\,i\,/\,ref\,i} = {}^{t}[\,M\,]_{/\,ref\,i}\ [\,I\,]_{components\,i\,/\,ref\,i}\ [\,M\,]_{/\,ref\,I}. \quad (28)$$

When the inertia matrix must be expressed in a different coordinate system than its original one, it can be translated in the same way

$$[\,I\,]_{components\,i\,/\,ref\,0} = {}^{t}[\,M_{(ref\,i->ref\,0)}\,]_{/\,ref\,i}\ [\,I\,]_{components\,i\,/\,ref\,i}\ [\,M_{(ref\,i->ref\,0)}\,]_{/\,ref\,I} \quad (29)$$

Once calculated, the system inertia matrix is used to build a single value objective function by calculating its norm. The choice of the norm depends on the type of objects present in the system. If the objects are isotropic (in shape) like a sphere or a cube, their inertia matrix is diagonal and invariant by rotation. In these cases, the norm, which consists of summing the squares of the diagonal terms, can be used.

$$\mathbf{Sqrt}\,[\,I_{11}{}^2 + I_{22}{}^2 + I_{33}{}^2\,]. \quad (30)$$

When off diagonal terms appear, the norm 1 or norm 2 can be used

$$\sum_{i=1..3} [\,\sum_{j=1..3} [\,I_{ij}\,]^2\,]. \quad (31)$$

Statically and Dynamically Balanced Loading

To obtain the static balance of a system it is only necessary to put its center of gravity at a given location. The center of gravity of a system is usually cheap to calculate and very accurate. The center of gravity of each item is calculated once when each object is

defined (shape and position). Then, when an object is moved, the same movement is applied to its center of gravity.

Statically balancing a system is sometimes not sufficient. A dynamic balance must be reached especially for objects that can move (cars, boats, planes...). The inertia matrix presents the interesting property of being a measure of the dynamic stability of a system. In order to be stable in dynamic situations as well as in static situations the principal axes of a system should respect some position. For example, the principal axes of the heaviest components of a car (underhood components) must stay aligned with the longitudinal axis of the vehicle.

Maintainability

The maintainability of a system is decomposed into two different tasks that must be performed and are represented by two objective functions.

Accessibility is the property of an object or part of this object to be seen and reached with the hand. This objective is implemented by a ray shooting algorithm in which the ray is an object simulating the volume necessary to access the component (Figure 61).

The removability of a component is defined by the amount of mechanical work to be done to remove it from the system (Figure 62). This mechanical work is proportional to the weight of the component. If some other components must be removed to remove a given component, the additional work is taken into account. The system maintainability is the sum of the accessibility of all the interesting components and of their removability.

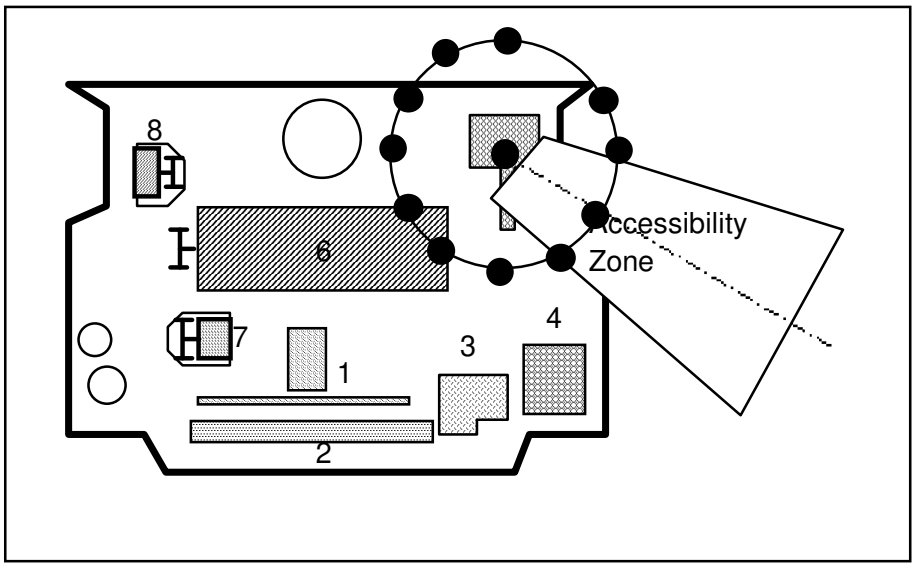

Figure 61. Accessibility. The accessibility is defined by the minimum size of the intersection between the accessibility zone and the components surrounding the object being accessed.

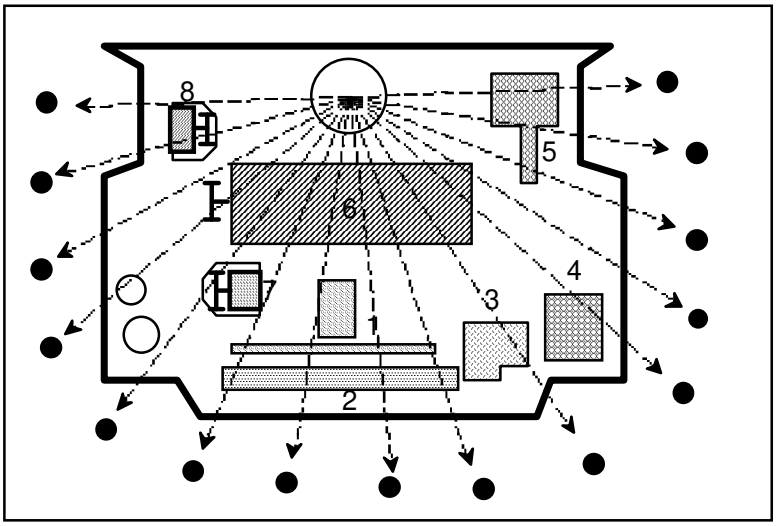

Figure 62. Removal path. Different possible removal path of the air filter in the 2D car engine. Each path is a composition of translations and rotations around the object center of gravity. In this case, some solutions would require to remove the motor (6) while others would require the removal of the power steering pump (8).

Both of these objectives were used as objective functions in the next chapter for testing the CDOM.

Interference Checking and Penalties

Checking interference between N objects can be done by a brute force algorithm which has a computational complexity growing like N^2, (N+1)*N / 2 to be exact. This decreases the performance of the evaluation of one configuration too fast when the number of objects increases. An algorithm using first level space partition into N equal size regions brings the average computational complexity down to N. Hence, using this partition, the intersection checking algorithm complexity grows linearly with the number of objects.

Conclusion

In this chapter, the implementation of the CDOM using multiple runs of a genetic algorithm in conjunction with populations of sets was presented. This approach provides a strong support for maintaining several solutions along the Pareto set. The advantage of relative placement and local search were assessed and the three strategies suggested in the hypotheses were analyzed.

A rank based objective function is defined as well as the combination objective/constraints through the penalty function. The ranking objective function plays the role of scalarization and thus allows the CDOM to deal with multiple objective CDPs.

Three engineering objective functions were chosen for the test cases, namely, compactness, center of gravity and maintainability. The objects are represented by their tessaleted surfaces extracted from a CAD package in STL formats.

In the next chapter, several CDPs are submitted to the CDOM after their complexity is assessed to validate or invalidate the hypotheses made in Chapter III and to bring support to the theoretical results given in Chapter IV.

CHAPTER V

TEST CASES AND ANALYSIS OF THE RESULTS

Introduction

From the initial strategy consisting in iteratively running a Genetic Algorithm and constantly narrowing the ranges of the variables around the best solutions found after each run, three methods are derived in order to test hypotheses one, three and four. Hypothesis 2 is tested by comparing the complexity classification based on Roughness and Dilution with the results of the test cases. In the first method, the components are placed relatively to a static component in order to reduce the possibilities of intersections. In the second method, the size of the solution is increased, as explained in Chapter III, by using a 0 order Local Search. Finally, the third method has the freedom to restrict or enlarge the ranges of the variables when either new solutions are discovered or when no improvement is obtained (Figure 63).

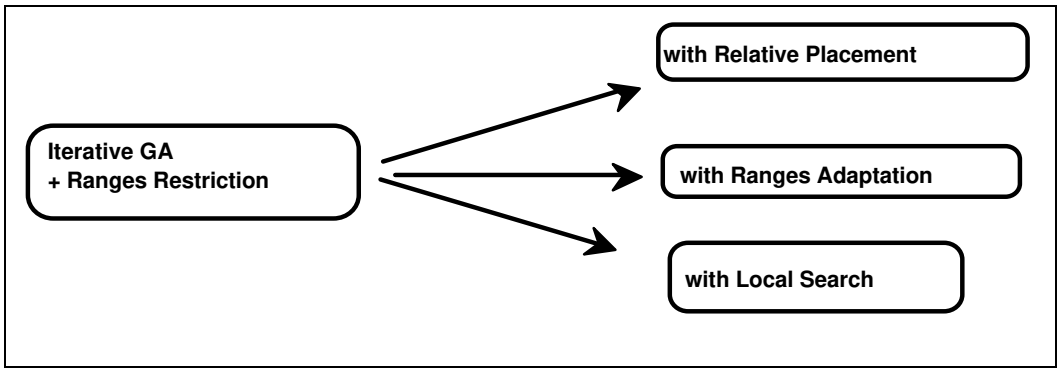

Figure 63. Proposed methods derived from the Iterative Genetic Algorithm with Range Restriction.

Several test cases involving a different number of components and variables are submitted to each method (Table XV). Each test case is rated in the classification proposed

in Chapter III according to the Dilution of the solutions, the Dilution of the feasible area and the Roughness of the landscape.

Table XV. Number of objects and number of variables for each test case.

Test Cases											
Objects	2	3 (2D/3D)		4	5 (2D/3D)		6	7 (2D/3D)		8	9
Variables	2	4	6	6	8	12	10	12	18	14	16

The study of several quality criteria, presented in detail in the next paragraph, allows to compare each method behavior. Eventually, the research hypotheses made in Chapter III are reconsidered under the light of the results obtained. The best method is then used to search for alternate solutions to the placement of components in a satellite and under the hood of a car.

Validation and Quality Criteria

Two types of criteria are needed for the analysis of the CDOM: criteria measuring the quality of the results obtained through CDOM, and criteria measuring the performance of the method itself. In the following, four criteria are presented for rating the resulting Pareto Set, namely: the distance of the extreme points from their references, the distribution and number of the points and the shape of the set. Three criteria are used to rate the CDOM itself, i.e. repeatability, generality and speed.

Solution Set Quality Criteria

The CDOM works on sets of points and expresses the solution of the Multicriteria CDP as a set which must be compared to the Pareto set. In practice this Pareto set is unknown. However, some characteristics of the solution were identified and translated into four criteria describing its quality.

Distance From the Extreme Planes

The distance between the solution set extreme points (EPs) and target extreme planes (TEPs) is useful for evaluating the quality of the solution set boundaries(Figure 64). The TEPs are either extreme planes indicated by the user, perpendicular to each objective axis

and corresponding to the optimum value of each objective function, or they can be found by solving an SOP for each objective of the CDP. The M extreme points of the solution set, M being the number of objectives, are determined in the objective space by comparing all the P points of each solution set $P_j(F_{1j}, F_{2j}, \ldots, F_{Mj})$ and retaining the extreme ones.

$EP_j(F_{1j},F_{2j},..,F_{Mj}) = P_j$ such that $F_{ij} = Min_{j=1..P}[F_{ij}]$.

Then, the criterion associated with each the solution set is

$$C_i = \mathbf{Dist}[EP_i, TEP_i] / (\mathbf{Max}[F_{ij}] - \mathbf{Min}[F_{ij}]) \quad (32)$$

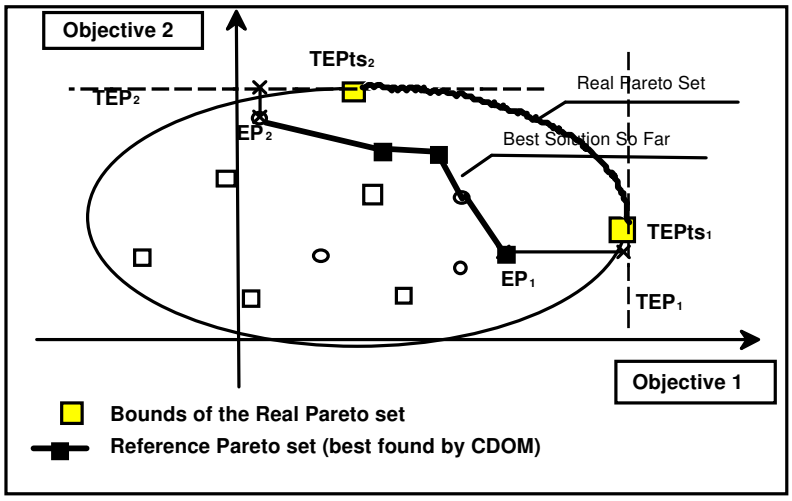

Figure 64. Extreme planes distances. Distance separating the Target Extreme Planes (TEPs) from the extreme points of the best solution found by CDOM (Eps). TEPs are planes perpendicular to one axis (in the objective space) and containing the Extreme Point (TEPts) of this same axis.

A comparison of these criteria with each others gives an idea of the relative difficulty of finding a solution for each SOP with respect to the others.

Point Spreading

In general the Pareto set of a bi-objective problem is either a curve or a set of curves. The output of the CDOM is a set of points which can be connected to one another in order to build an approximation of this Pareto Set. This chain establishes an order between the points

and thus enables the measure of the distance between two consecutive points (Figure 64). The mean and standard deviation of the distances separating all the consecutive points gives a measure of the spanning of the solution and of the distribution of the points. With a higher number of objectives, the Pareto set can be either one or several curves, or one or several surfaces. In this latter case the notion of consecutive points is lost and thus, even if the distances between points can be calculated, the order of the points remains to be found which is known to be a complex task and which is not the aim of this work. For this reason a criterion independent from the order of the points on the curve, surface or hyper surface was chosen, in which the distance separating consecutive points is replaced by the number of neighbors contained inside a spherical region around each member of the solution set (Figure 65). This choice has the additional advantage to still be applicable on non continuous sets. The study of the standard deviation of this number provides a measure of the even distribution of the points. The closer the standard deviation to zero the better the distribution. However, this criterion must be coupled with the study of its Average and with the previous criterion to be able to detect clustered solutions in which all the points are grouped into a very small area.

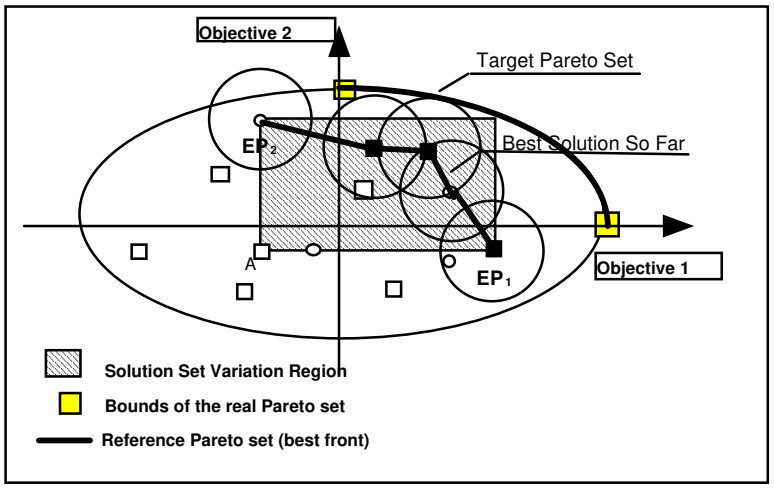

Figure 65. Point spreading control.

The only unknown to be determined before calculating the criterion is the radius of the spherical regions. If this radius is too small the number of points inside each region will be one and still the distribution might be uneven. If the radius is too large each region will regroup all the points of the set whatever their distribution. The ideal solution to this problem in 2D is given by the exact length of the solution set divided by the number of points in this same set minus one. Although this evaluation is easy for 2D curves, it is a completely different matter in 3D since the solution surfaces can have any shape ranging from 0 area to the three sides of a cube. If the area of the 3 objective solution set could be easily measured, the radius of the sphere would have been given by:

$$r = \mathbf{Sqrt[}\ (\text{Solution Set Area})/\ (\Pi^*(P-1))\ \mathbf{]}. \tag{34}$$

where P is the number of points in the solution set.

In spite of the fact that the Solution Set Area is unknown, it is possible to bracket its value between two bounds. For the 2D case these bounds are

$$0 <= \text{Length Pareto Set} <= \mathbf{Dist[}\ EP_1,\ A\ \mathbf{]} + \mathbf{Dist[}\ A,\ EP_2\ \mathbf{]} \tag{35}$$

where A is the corner of a rectangle which has the Extreme Points EP_1 and EP_2 as opposite corners Figure 65.

For the 3D case the process is the same and the maximum area of the solution set is estimated to be half the sum of the sides area of the box having EP_1, EP_2 EP_3 and A for corners (Figure 66).

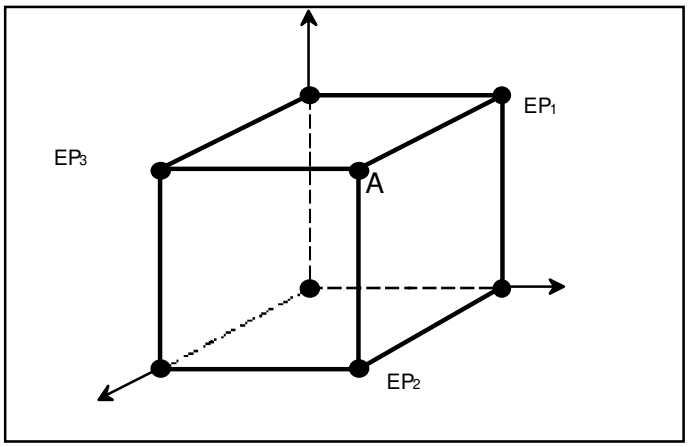

Figure 66. Bracketing the area of the Solution Set in 3D.

With this area it is possible to evaluate the radius of the spherical region which serves as basis for calculating the distribution of the points in 3D.

$$r = \mathbf{Sqrt[}\ (\text{Maximum Solution Set Area})/\ (\Pi^*(P - 1)\ )\ \mathbf{]}. \qquad (36)$$

where P is the number of points in the solution set.

Shape

The distribution of the points is a "structural information" which gives clues about the internal characteristics of the solution. Measuring the distance of the EPs from the best solution is not a sufficient information to position the final cloud since it only involves the extreme points positions. Hence a third criterion was added in order to grasp the "global shape" of the final set and to compare the solutions between themselves.

Similarly to a regression coefficient in statistics which is used to assess the quality of a fit using the sum of the squares of the distance between measured data and a regression curve, a measure of the distance between the solution set points and a reference plane is used. Ideally, this plane would have been the plane defined by the three TEPts, however, once again since these points are unknown, another reference is taken. The TEPts can be placed anywhere along constant lines or planes (or hyperplanes) perpendicular to the axis corresponding to their extreme objective value i.e., on Figure 64, TEPts1 can be located anywhere on a vertical line as long as it stays in between Max[Objective 2] and Min [

Objective 2]. Hence, the reference chosen is the diagonal of the objective space (Figure 67). Then, the sum of the distances between the solution set points and this reference is calculated and gives a number that can be used for comparison betweens runs.

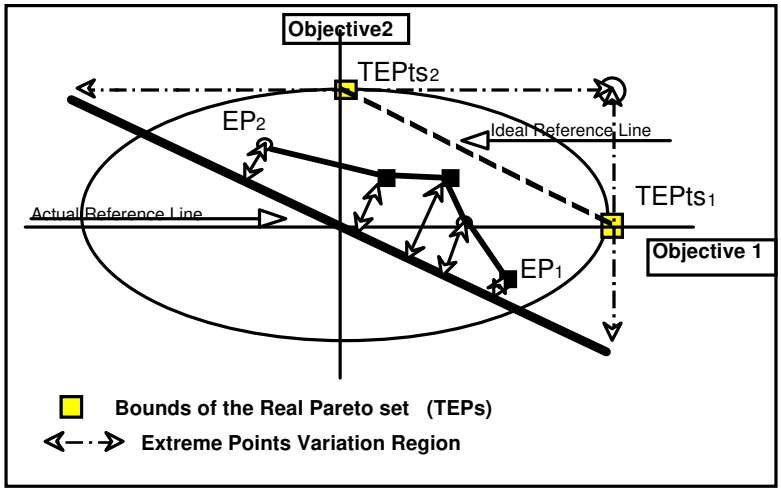

Figure 67. Third criterion references.

From these measures, two criteria can be studied. First, the standard deviation of the average distance indicates the ability of the method to provide the same set run after run and second the standard deviation of the standard deviation of the distances indicates the ability of the method to provide the same shape of the front.

Method Quality Criteria

Repeatability

Since the core of the CDOM relies on a non-determinist search algorithm, the results obtained for a given test case have a high probability to be different each time the method is run, especially because the CDPs are non-linear problems involving many variables. In order to measure the degree of confidence provided by a single run, each test case is submitted 5 times. Each criterion defining the solution quality, i.e. the distance from the TEPs on each axis, the point spread and the shape, are recorded. The standard deviation of these measures over the same test case indicates the repeatability of the CDOM for a given CDP while the average estimates the average results quality that can be expected knowing that the best

distance from the TEPs is zero. Since the ideal shape factor is unknown, nothing can be concluded about the quality of the results from this factor; only its standard deviation is useful and states the ability of the method to provide the same shape when applied a second time on the same test case.

Generality

All the CDPs presented belong to the same family i.e. they have the same objective functions. The differences presented in the test cases only concern the objects. They involve an increasing number of objects with same shapes and size. This allows the identification of the difficulties coming from the size of the CDP.

The Generality of the CDOM is established by comparing the results over the full span of the test cases.

Speed

The CDOM relies on an iterative use of a genetic algorithm. Each run of the GA is executed until convergence is reached. According to the strategies, the global iterative process of CDOM is stopped when the reference Pareto set cannot be enriched further i.e. when no new point is discovered during 2 consecutive runs of the GA.

Since usually the CDOM spends its time in the objectives calculation, the number of function evaluation is chosen as Speed measure.

Academic Examples

The academic examples rely on test cases for which the solution is known. Cubes were chosen because of their sharp edges, flat surfaces and very compact solutions for the packing problems which make these CDPs better handled by combinatorial methods when dealing with removability issues. However, balance and compactness issues are better handled by continuous non-linear optimization methods. Each example is described in terms of variables, objects shape and objective functions. Its position in the classification is established after the Dilution of its feasible area, the size of its solution and its roughness are calculated. Then, the quality criteria statistics are presented and analyzed.

Cubes Configuration Design Problems

The cube CDP is extracted from the well known packing problems reviewed in chapter II (Square into square container). It was augmented by the addition of two objectives which are center of gravity and maintainability.

Problem statement

In this CDP, N cubes must be placed in space such that the compactness, position of center of gravity and maintainability are optimized. Two versions of this problem are studied: a 2D version in which the cube displacement is restricted to translations tx and ty (Figure 68); a 3D version in which all the translations are authorized. The ranges of the translations correpond to the size of the container.

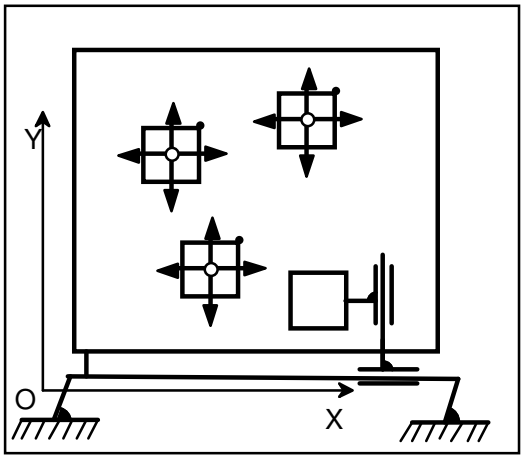

Figure 68. In the 2D version of this CDP the cubes can translate along the x and y axis.

The initial problem statement, which consists in optimizing the volume of the bounding box enveloping the cubes, was changed in order to match mechanical objective functions. Thus, the compactness of the configuration is measured by the inertia matrix of the system and two more objectives are added to the problem in order to build the CDP: distance of the center of gravity from a target point (center of the initial problem space), and maintainability which specifies that the cubes must be removable from the system through the right side (Figure 69). In order to remove many configurations obtained by translations one cube is fixed at (0,0) in order to get a non singular Pareto Set (otherwise the maximum

compacity can be reached at the same time as the coincidence of the Center of gravity which makes the Pareto Set degenerate into a single point). All the cubes have the same weight (10) and volume (20x20x20).

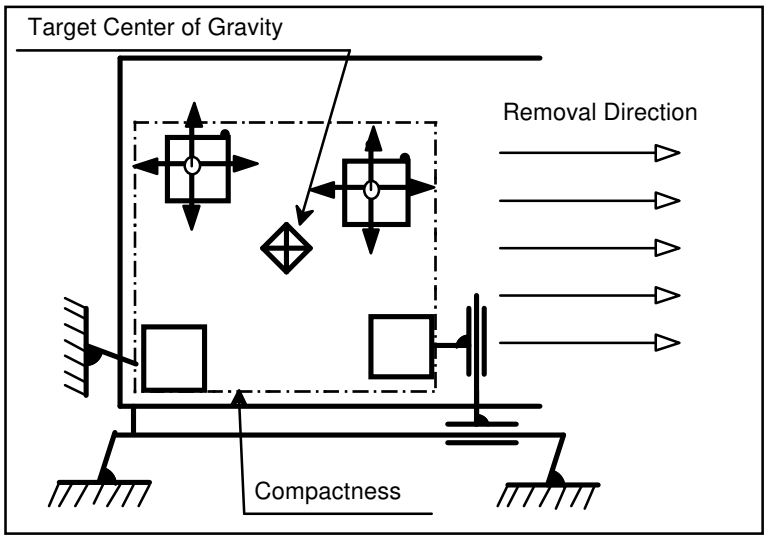

Figure 69. Objectives of the cube CDPs.

Since each test case must be put into similar initial condition, the size of the initial container i.e. the arena inside which the objects will move is calculated such that the dilution of the Feasible Area is 1/3 which gives, in the case of the cubes, the following expressions:

$L2D = (3 N 400)^{1/2}$.

$L3D = (3 N 8000)^{1/3}$.

in which N is the number of cubes. These length are given in Table XVI for each test case.

Table XVI. Initial data for the 2D and 3D cube CDP. L2D and L3D are the respective sides of the square and cubic container in which the cubes move.

N	2	3	4	5	6	7	8	9
L2D	49	60	69	77	85	92	98	104
L3D	--	42	--	50	--	56	--	--

The only 3D test cases interesting are those for which an equivalent variables space size can be found in 2D, hence the 3, 5 and 7 objects 3D tests cases were the only ones involved in this study.

Complexity Measures and Classification

From the specifications given in the previous paragraph and the definitions of each criterion proposed in Chapter III defining the complexity of a CDP, the Dilution of the feasible area and solutions, and the Roughness can be calculated for each test (Table XVII).

Table XVII. Complexity criteria for each cube test case.

Number of Objects	2	3 (2D/3D)		4	5 (2D/3D)		6	7 (2D/3D)		8	9
Dil_{FA}	1/3	1/3	1/3	1/3	1/3	1/3	1/3	1/3	1/3	1/3	1/3
Rg (log)	0.7	1.7	1.7	3.2	4.8	4.8	6.6	8.5	8.5	10.6	12.8
Dil_{Sol1}(log)	-5.7	-11	-17.5	-17.2	-22.3	-25.7	-27.9	-33.1	-42.4	-38.3	-43.4
Dil_{Sol2}(log)	-13.8	-10.4	-10.5	-8.5	-6.9	-7.5	-6.3	-5.7	-5.9	-5.3	-5
Dil_{Sol3}(log)	-.5	-1.5	-0.8	-3.8	-2.8	-2.3	-3	-6.9	-4.1	-9.2	-9.4

Since the Dilution is the same, the 3D classification described in Chapter III can then be represented in 2D (in the plane Dil_{FA} = 1/3) for each objective function. This classification is used as reference to explain the results obtained through the use of the different methods.

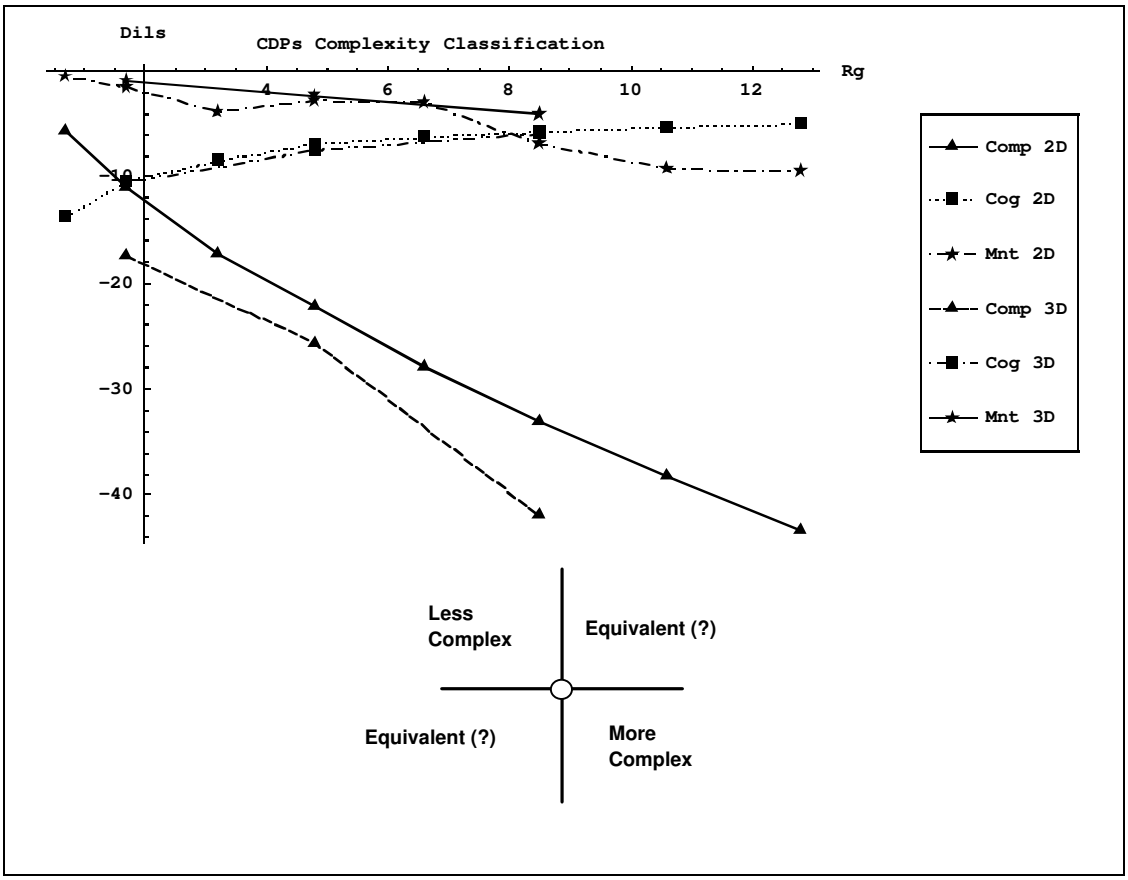

Figure 70. Complexity classification of the cubes CDPs test cases. Each point represents one cube CDP. Roughness is indicated on the horizontal axis and dilution on the vertical axis. This graph represents a cut at Dil_{fa} = 1/3 of the classification proposed in Chapter III (Figure 42).

All CDPs rated in the upper left quadrant with respect to a reference problem are considered as less complex than the reference while all CDPs belonging to the lower right quadrants are more complex. The problems lying in the other quadrant are considered as having an equivalent complexity with however an uncertainty.

Two features of this classification must be retained. First, the difference in complexity of 2D and 3D CDPs involving the same number of variables. Three dimension CDPs are easier to solve due to the fact that, first, they involve less objects for the same variable space size, and second, their solutions domain are more extended than their two dimension equivalent CDPs. Second the scales on each axis correspond to natural logarithms, which implies that for each unit the complexity is multiplied or divided by 2.7.

Results Presentation and Analysis

The final result that is of interest for the engineer is the Pareto set in the objective space and its corresponding values in the variable space each of which reflects a configuration in 3D space (Figure 71). Extreme points of this final set represent "extreme" configurations corresponding to the solutions of the Single Objective Optimization problems involving each objective alone.

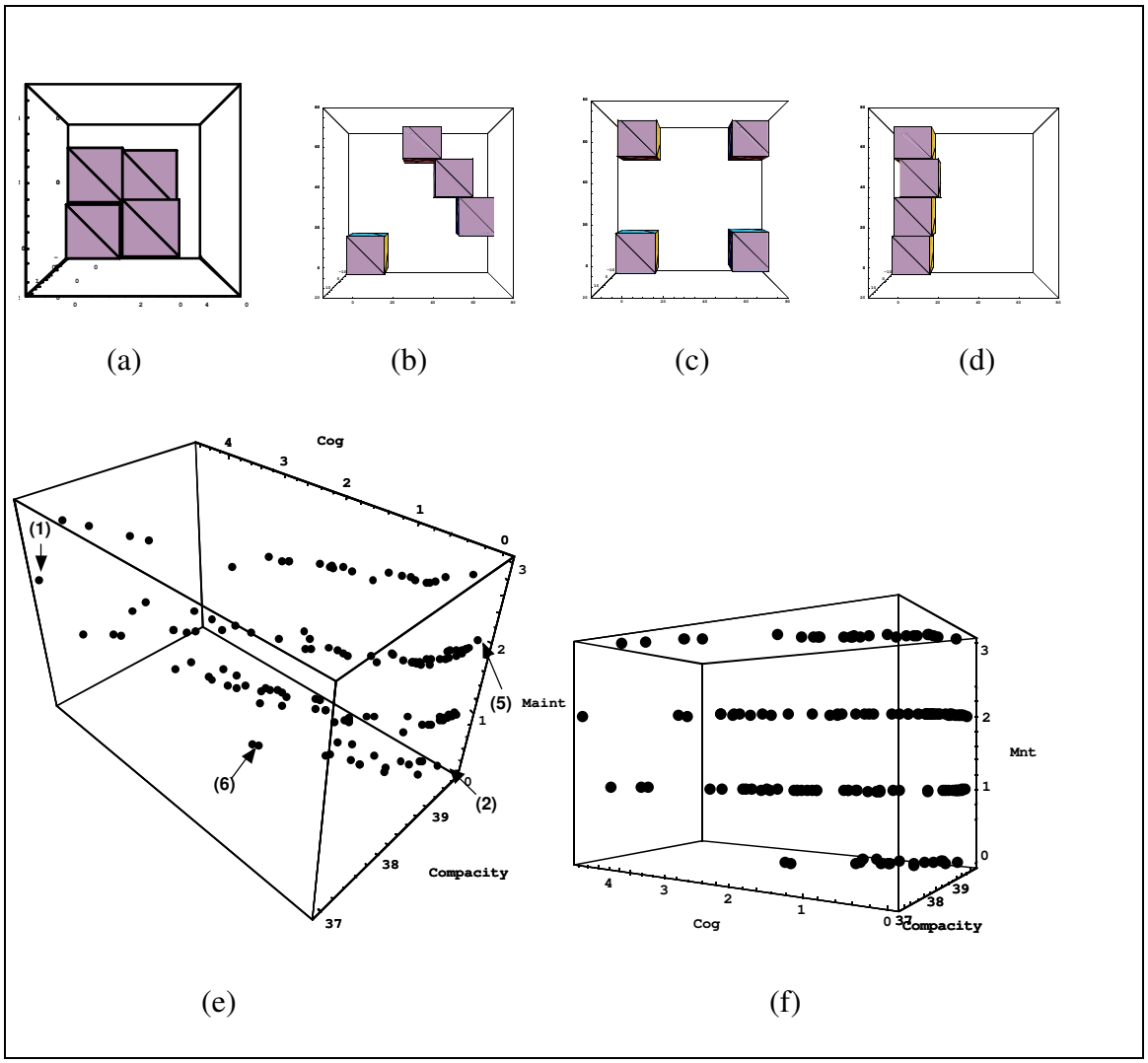

Figure 71. Example of final set distribution and configurations. Each extreme point of the final set (e and f) in the objective space corresponds to an extreme configuration (a, b, c and d) in 3D space. Maintainability (Maint) is indicated on the vertical axis. Distance from the target center of gravity (Cog) and Compacity are indicated on the horizontal plane.

As expected, the final solution is made of layers of points. Each layer corresponds to a constant maintainability (Figure 71). Within each layer, the solution of the bi-objective optimization problem involving Center of Gravity and Compactness appears as a sampled curve (Figure 72).

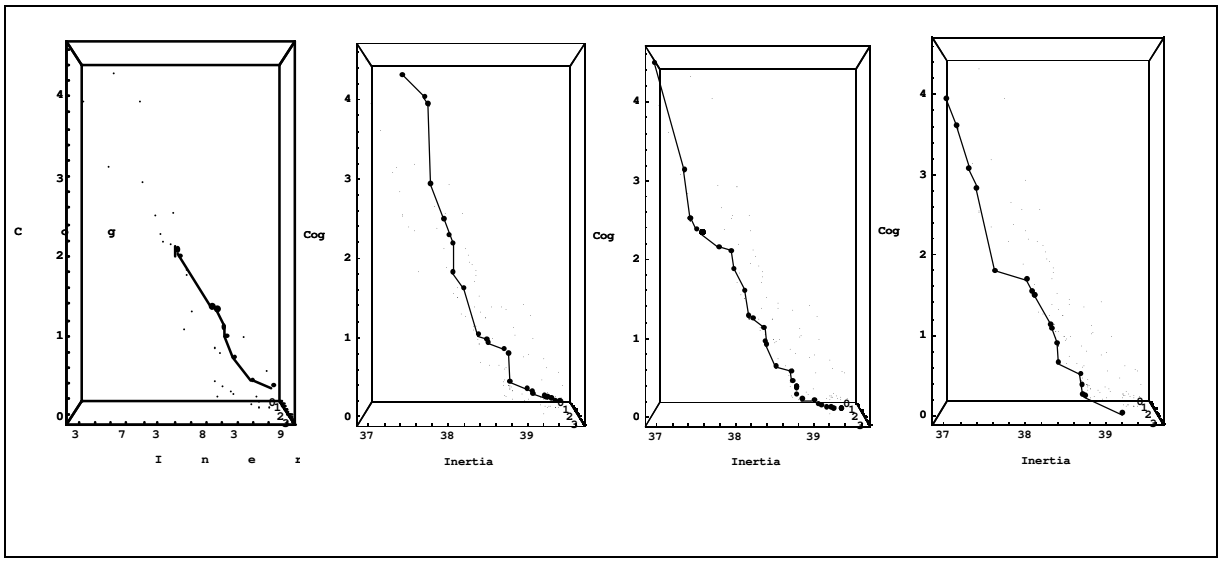

Figure 72. Three different layers of the 4 cube CDP. From left to right, Maintainability = 0, Maintainability = 1, Maintainability = 2, Maintainability = 3.

It can be noticed that, as the system becomes less maintainable, the curves translate toward the left indicating that a better compactness can achieved.

The same type of solution set is studied in the following, for all the test cases and for the four methods investigated i.e. :

- Ranges Restriction (R), which is the algorithm used as reference and already tested on Multicriteria non-convex optimization problems by Grignon and Fadel (1997).
- Relative Placement (RP), which consists in placing the moving cubes with respect to the static one (Figure 69) and to restrict their possibility of movements.
- Range Relaxation (RLX), which implements the idea of enlarging the search area whenever no solution is found within a region of the variable space.
- Local Search (LS), relies on a zero order Local Seach to extend the size of the solution and of the feasible area in variable space.

The graphs that serve as a basis for the conclusion have the number of variables of the CDPs along the horizontal axis and the criteria along the vertical axis.

Distance from Target Extreme Plane 1

The distance from the first target extreme plane criterion, TEP1, is plotted using a logarithmic scale of the system inertia matrix norm (equation 30) scaled using equation (32) such that the results correspond to percentages of the range of variation of this objective function. Hence, an increase of approximately 25% on the result graph represents a doubling of the system inertia.

Methods Comparison. On one hand, Iterative GA with Local Search (LS) and Iterative GA with Relaxation (RLX) both perform better than the base strategy (Iterative GA with Range Restriction). On the other hand, the Relative Placement strategy (RP) is performing nearly consistently worse than the reference method (Figure 73 top). This latter result is in contradiction with what was expected. This behavior repeats itself in the 3D test cases (Figure 73 bottom).

The Local Search provide the best results for the best compactness search.

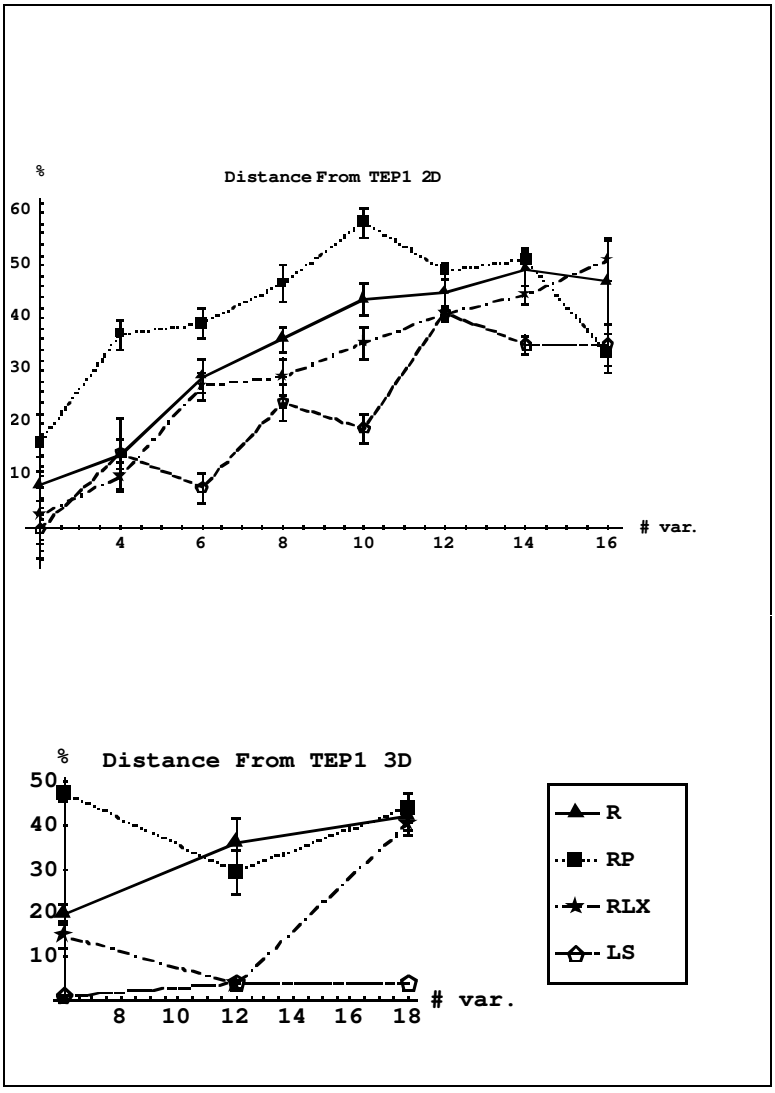

Figure 73. Distance from TEP1 achieved by the four proposed algorithms.

Two dimension versus Three dimension CDPs Comparison. The 3D test cases were consistently easier to solve than their 2D counterparts with the same number of variables (Figure 70). This result corroborates the classification based on the Dilutions and Roughness (Figure 74).

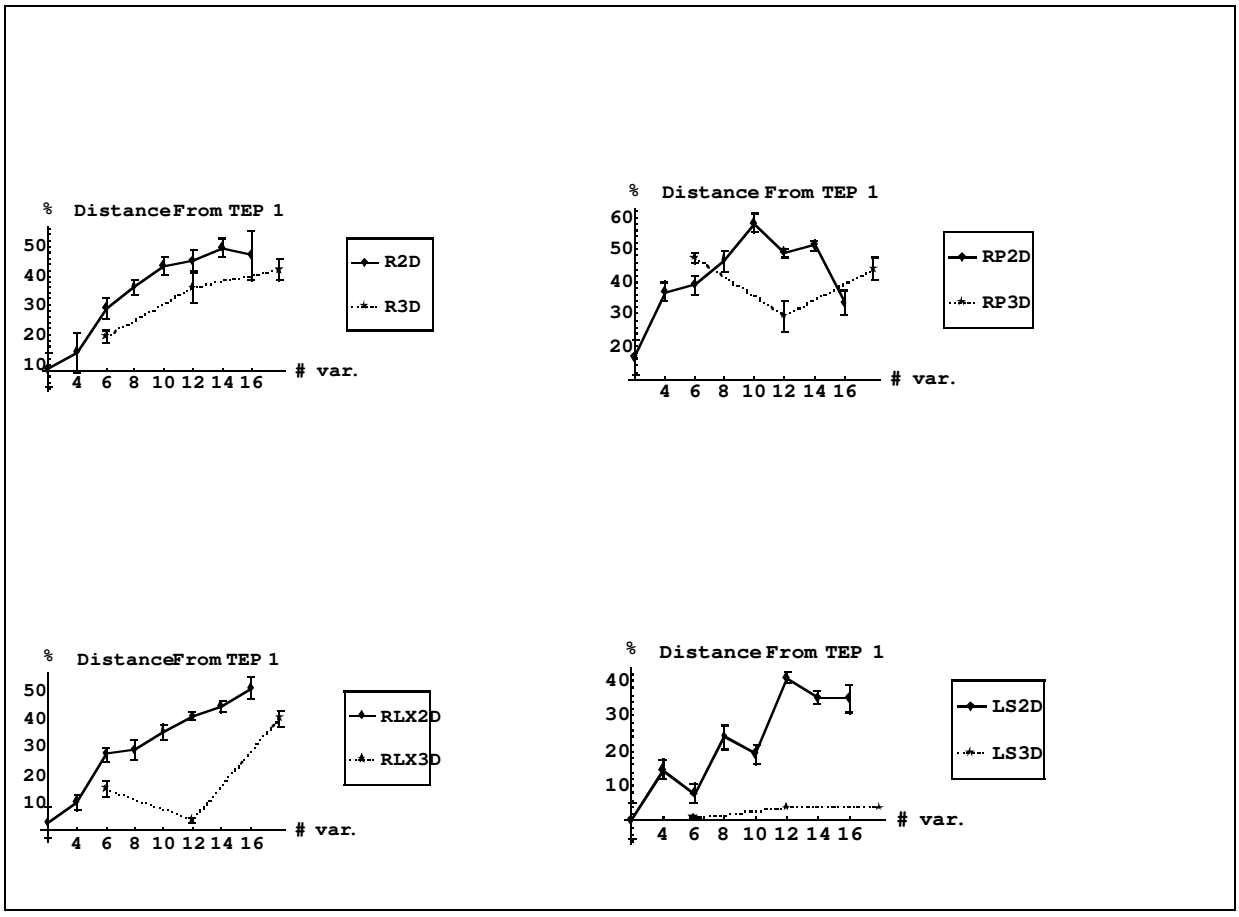

Figure 74. Comparison of the distances from TEP1 for 2D and 3D test cases.

General Remarks. The overall behavior that was expected was a degradation of the precision while the number of variables increased. The only method that does not obey this rule is the Relative Placement strategy which shows an improvement for cases with more than 10 variables. One of the explanations for this behavior could be that the restriction of the middle size test cases ranges (4,6,8 and 10 variables) was too drastic and thus removed many solutions from the variable space.

Distance from Target Extreme Plane 2

The distance from the target extreme plane 2 criterion, TEP2, represents the distance between the system center of gravity and the center of the square or cubic container whose side length was defined in table XVII (Figure 69). It is expressed as a percentage of the maximum value that the CDP under study can reach.

Methods Comparison. All the methods exhibit an improvement with respect to the reference algorithm (R) in a large majority of the test cases. This conclusion is valid from the point of view of the precision and also from the point of view of the stability (presence of small error bars)(Figure 75 top). The same conclusion is achieved for the 3D test cases (Figure 75 bottom). It appears that there is no significant change in the performance of each method while the number of variable increases. The CDP Complexity Classification indicates that the complexity of the 2D examples might be nearly constant. This is due to the fact that when the variables space size increases, the size of the solution increases as well.

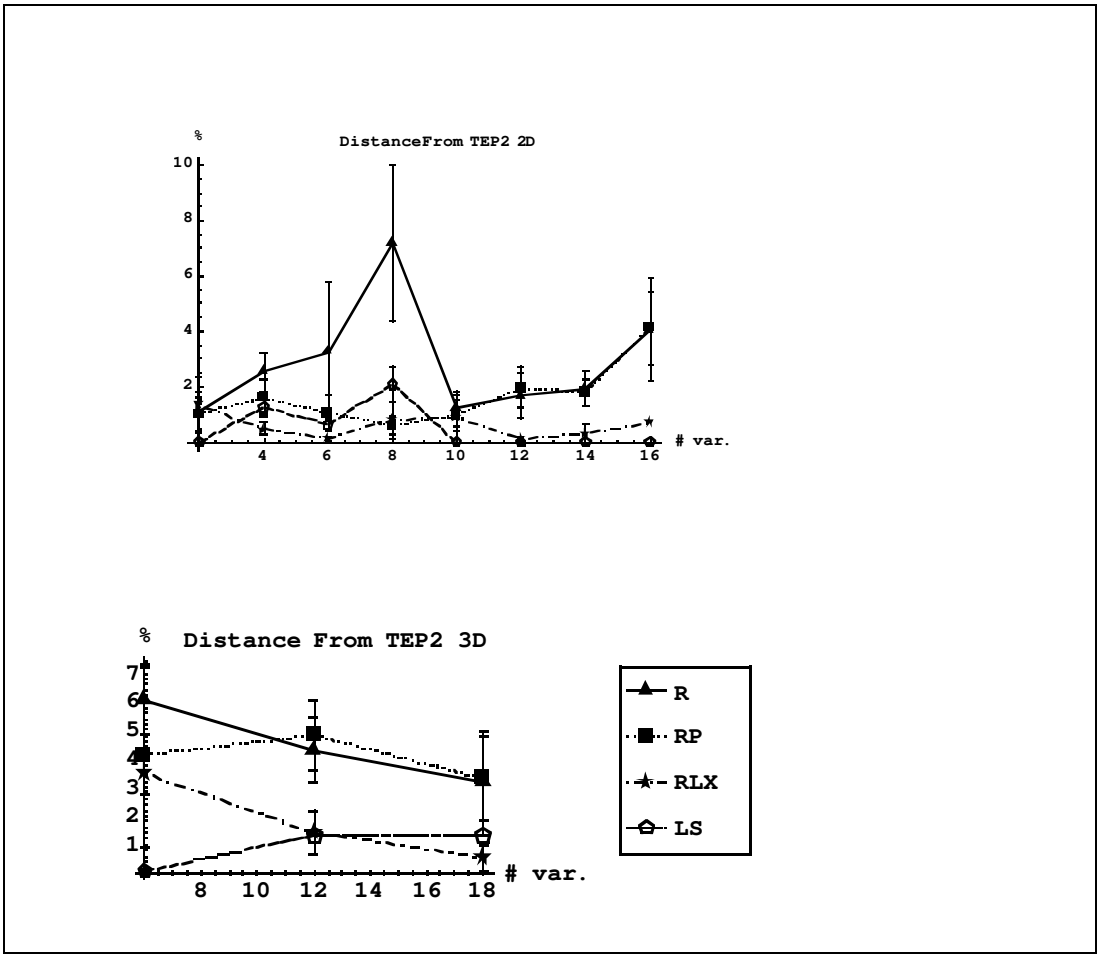

Figure 75. Distance from TEP2 achieved by the four proposed algorithms.

Two Dimension versus Three Dimension CDPs Comparison. Contrary to the Compactness results, the 3D center of gravity results are consistently worse than their 2D counterparts. The difference is small (around 2%). This result is in agreement with the classification which exhibits 2D and 3D center of gravity complexity curves nearly confounded (Figure 70, Cog 2D and Cog 3D curves). However, it seems difficult to say if such a small difference in complexity justifies the 2% gap between 2D and 3D results.

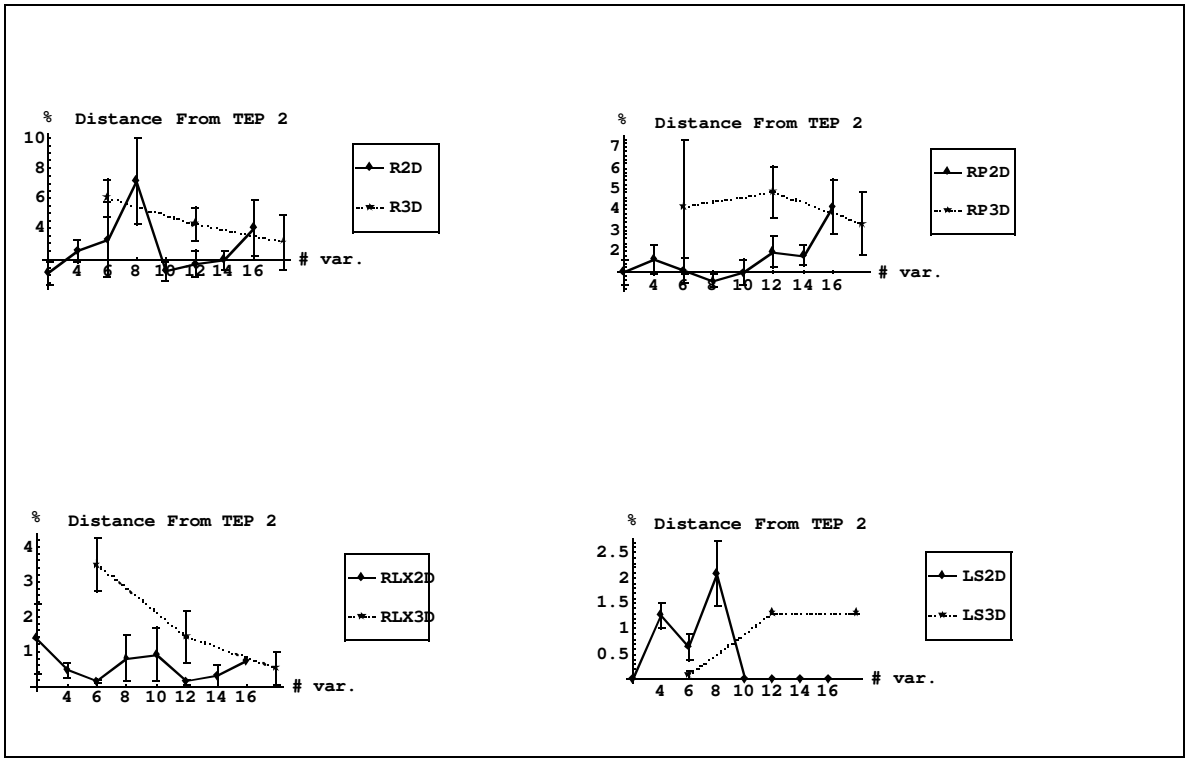

Figure 76. Comparison of the 2D and 3D distances from the target center of gravity.

General Remarks. The precision achieved by all the methods on all the test cases is better than the results obtained on the Compactness (distance from TEP1). All the results are within 8% (of the maximum range) of the target with an average around 2%. This better performance, attributed to the higher number of solutions for the center of gravity with respect to the compactness, is correctly predicted by the classification (Figure 70).

It must also be noted that there is no decrease in performance while the number of objects increases. A possible explanation is that the number of solutions grows at a rate that

counterbalances the increase in complexity of this CDP. The Complexity classification confirms this version; the complexity of the Center of Gravity CDP seems only due to the Roughness.

Distance from Target Extreme Plane 3

The Third Target Extreme Plane represents the best removability for the considered CDP. TEP3 expresses the removability of the studied CDP as a percentage of the range of possible values for this criterion.

Methods Comparison. On average, all the methods but the Local Search present an improvement over the reference for the 2D test cases. The best method is RLX for the 2D test cases and LS for the 3D although RLX comes close (Figure 77).

Except for the Local Search these results are consistent with the classification (Figure 70). An explanation was found for the deviation of the LS algorithm. The method chosen (0 order Brent Local Search) minimizes the penalty function, along each coordinate axis separately starting from the x-axis. The search is stopped as soon as the minimum in one direction is reached. Hence, an overlapping object will slide first along the X direction until the penalty vanishes, then the compactness is taken into account in order to prevent the object from wandering off. This privileges configurations in which the components are staked along the X direction and thus results in a bad removability.

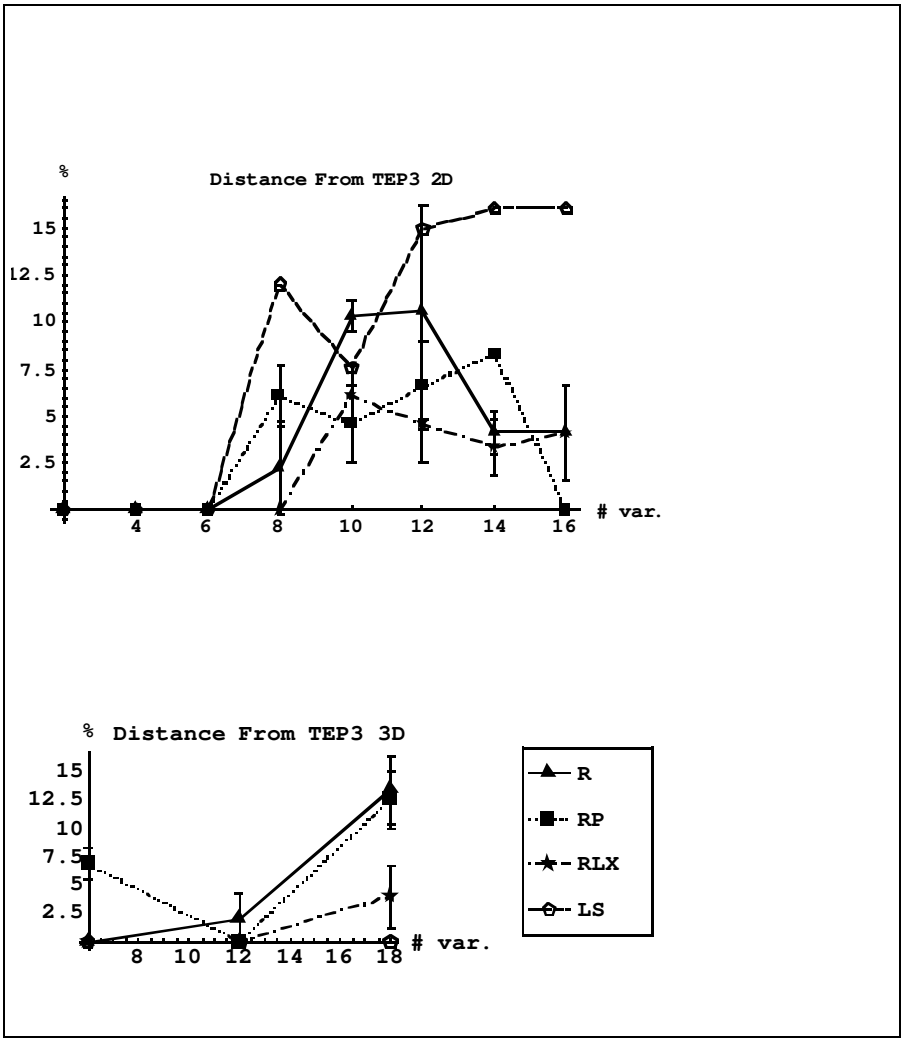

Figure 77. Distance from TEP3 achieved by the four proposed algorithms.

Two Dimension versus Three Dimension CDPs Comparison. The results indicate the 3D problems are slightly easier to solve. This small difference is confirmed by the Classification.

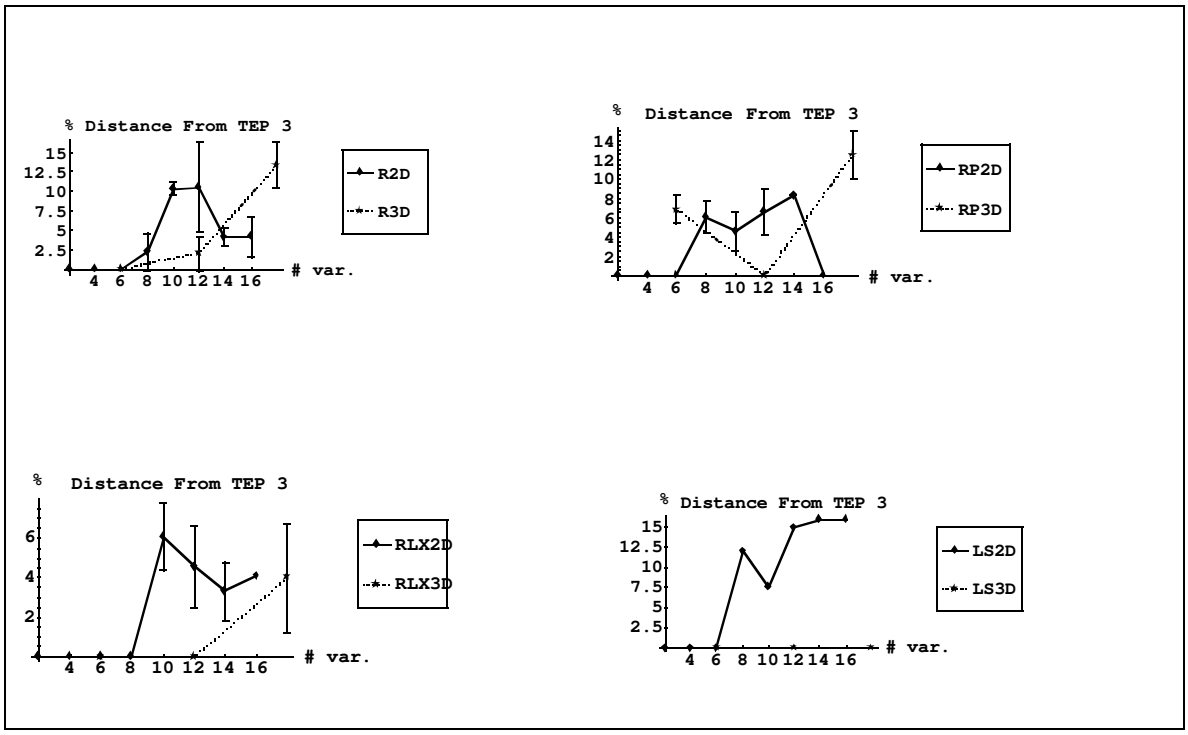

Figure 78. Comparison of the 2D and 3D distances from the target center of gravity.

General Remarks. Once again, the RLX method outperforms the other algorithms with an average of 2% (of the maximum solution range) away from the target solution. However, all the results do not present the steadiness exhibited by the previous criterion (TEP2) (see oscillations and size of the error bars on Figure 78). This phenomenon can be attributed to the fact that, Removability is a discrete function for which the number possible values is small. Hence, a jump of one unit along this axis represents a large variation of the total Removability. An interesting remark concerns the oscillations of the quality of the results. This behavior might also have been foreseen by closely looking at the classification curves for maintainability and raises the question of the relative complexity of one objective with respect to the others. This is debated in Chapter V.

Global Comparison of the Methods for the Distance from the Extreme Planes.

The analysis of the distance evolution suggests that the points cluster in a small portion of the Pareto set at the expense of the compactness objective (Figure 73, 75, 77). The solution shrinks along the compactness axis while remaining equally spread along the center

of gravity axis and while fluctuating on the Removability axis. This behavior correctly reflects the complexity classification of the test cases i.e. the methods have more difficulties to find optima of the more complex Compactness objective function and find equally good solutions for the Center of Gravity. The quality of the maintainability results fluctuates as well as their complexity.

Based on the first three criteria, ranks are attributed to each method (Table XVIII) that allow to rate their performance with respect to each other. The RLX and LS methods present the best improvements with respect to the base strategy.

Table XVIII. Classification of the methods based on the distances from the target extreme planes criteria.

	R	RP	RLX	LS
TEP1 2D	3	4	2	1
TEP1 3D	3	3	2	1
TEP2 2D	4	2	1	1
TEP2 3D	2	3	1	1
TEP3 2D	3	2	1	4
TEP3 3D	4	3	2	1
Total	19	18	9	9

In addition to this classification, already two technical rules seem to emerge for constructing a better search:

- The Local Search employed for finding feasible solutions must be insensitive to the order in which the search directions are chosen in order not to introduce a bias in the final configurations.
- The range restriction removes solutions from the problem and thus reduces the efficiency of the method. It must be used with components that are not permutable in order to keep as many solutions as possible in the variable space. It is reminded that permutations bring a factorial number of solutions. This drawback should disappear when the relative placement is applied to enforce meaningful constraints, in mechanical assemblies for instance, between components that are, in most cases, different in shape and weight. This was not the case in the test cases investigated.

Criterion 2 Analysis

Criterion 2 standard deviation is a measure of the even distribution of the points. The closer to 0, the more evenly distributed the points are. Criterion 2 average measures the density of points in each spherical region. It must be combined with the Distance from the

TEPs and the number of points in order to make the distinction between extended Pareto sets containing many points and small sets containing few points.

Methods Comparison. It seems difficult to identify a clear domination of one method from the analysis of criterion C2. The density of points in the solution set drops as complexity increases. This drop is regular for the RLX and RP methods and it is less regular for the Local Search (LS) (Figure 79). On average, all the methods, LS excepted, perform better than the base strategy. However, the RLX method provides a slight advantage over its competitors in the majority of the cases.

The study of the standard deviation (DEV c2, Figure 79) gives an insight of the evolution of the points distribution inside the solution set. The drop in standard deviation, which means an unexpected improvement of the distribution of the final set, can be explained by the fact that, on one hand the number of points is reduced (Figure 80) and, on the other hand the solution set clusters in a small area of the Pareto set. A reduction in size along one of the continuous objective function axis implies that the influence of the discrete objective function (removability) on the points distribution increases. Since the curves of equal removability are equally spaced in the objective space, the points are automatically better distributed. Hence, the distribution improvement is not the effect of the methods themselves.

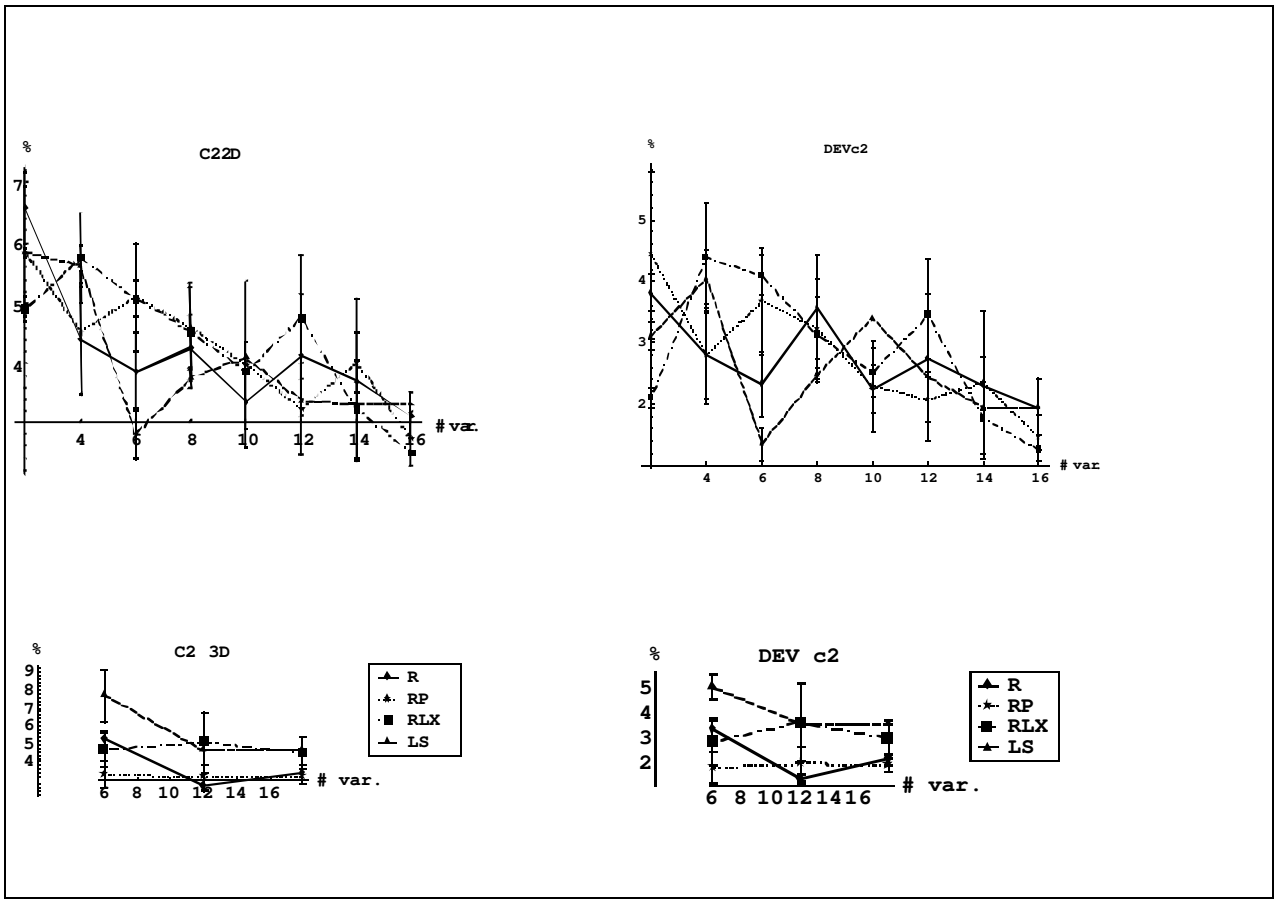

Figure 79. Criterion C2 evolution in function of the number of variables.

C2 average can also be compared with the evolution of the number of points discovered by each method (Figure 80). As the number of objects increases, the classification shows that the complexity of two objectives out of three increases, the third one remaining constant (Center of Gravity).

Although it is not a proven fact, it is likely that the relative size of the Pareto set shrinks accordingly. This would explain that the methods have more and more difficulty to discover new Pareto points.

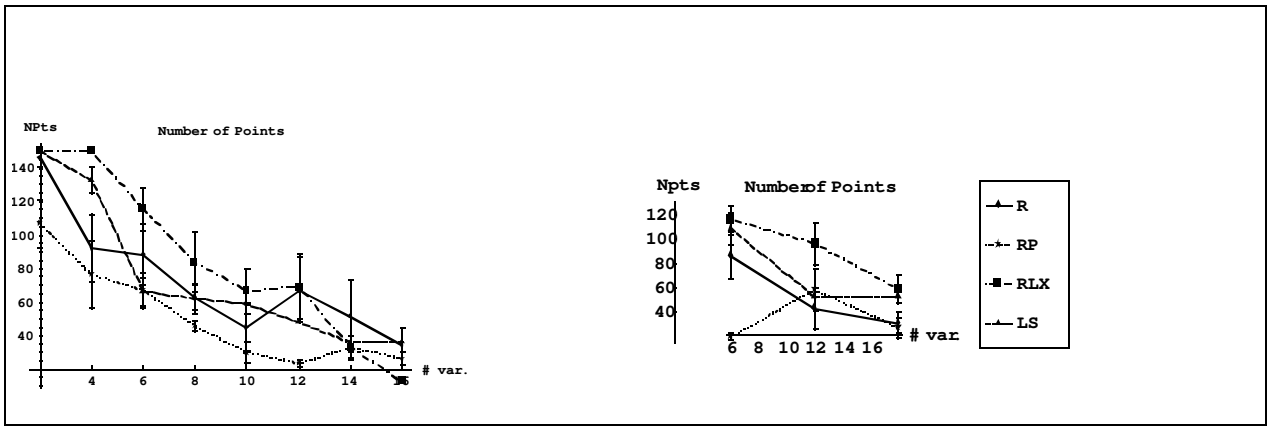

Figure 80. Comparison of the 2D and 3D distances from the target center of gravity.

Criterion 3 Study

For this criterion, indicating the average distance between the solution set and a reference plane (Figure 67), the further away from 0 the values are, the better the method performs. The average of the standard deviation of the distances indicates the flatness of the set. The smaller the standard deviation the flatter the final solution is.

Methods Comparison. All the methods outperform the initial strategy in 2D and 3D, the best one being by far the Local Search with one disadvantage coming from the large standard deviation implying that the front is either jagged or tilted with respect to the reference plane. A visual inspection of the solution sets indicates that the average distance reflects accurately the real state of the points. On the contrary, the flatness criterion (Dev c3) does not bring enough insight in the real shape of the cloud since some solutions might be very "flat" but not parallel to the reference plane and thus exhibit a high standard deviation.

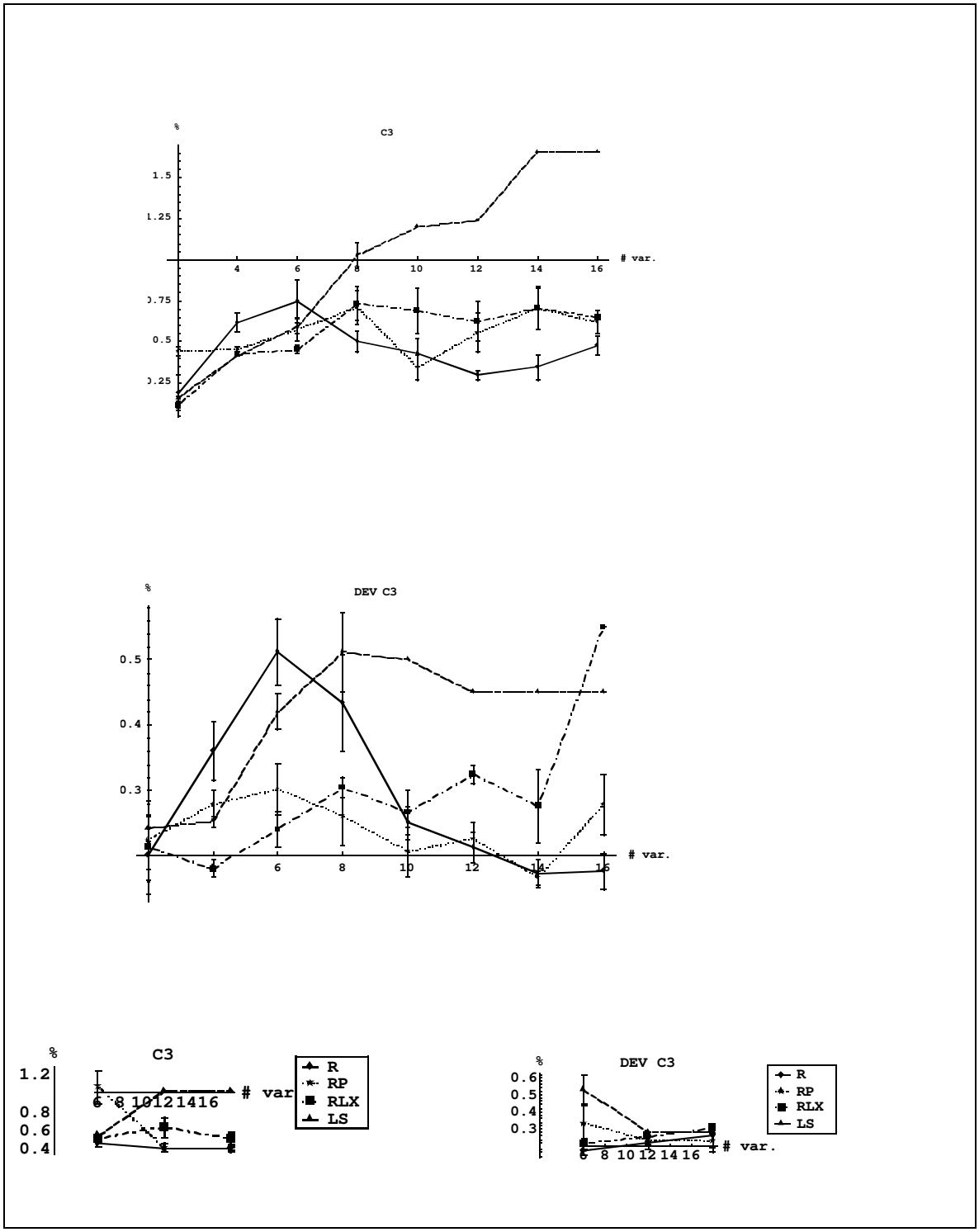

Figure 81. Distance from the reference plane average and standard deviation of the distances from the reference plane.

Summary On the Set Quality Measures. It seems that no method is able to provide a clear advantage over the others for obtaining a flat well spread Pareto set containing evenly

distributed points. This might be due to the fact that the distribution of the points in ensured by an additional term of the Rank Objective function which leaves the priority to the discovery of Pareto points (see Equation 23).

Speed Measures: Objectives and Penalty Evaluations

The cost of all the methods is assessed by the number of function evaluation needed to reach convergence. The RLX method has the highest cost which was expected considering the nature of this strategy (Figure 82). It was also expected that the number of penalty functions evaluations would be higher than R and RP for this same method. The exponential slope exhibited by the curve indicates that this method might not be scaleable. The cost of the Local Search was expected to be low but not by that much. However, only the number of penalty functions evaluations that trigger the local search are reported here. This number must be multiplied by 20 times the number of objects on average to take into account the evaluations done during the local search routine. This makes the LS algorithm computationally much more expensive than the RLX algorithm.

The only difference of the RLX with respect to the reference method (R) is the possibility to enlarge the search area. No additional advantageous information is provided to this method with respect to the information given to the other methods. It seems then normal that it requires more penalty evaluations before stopping.

The smaller number of penalty function evaluations of the LS search (without taking into account the evaluations performed by the local search routine) provides a confirmation to the hypothesis 1 of this work about the bias introduced by local minima based populations. This method was indeed designed to introduce more feasible points in the population of the Genetic Algorithm.

The global comparison of the number of function evaluation points out a slight decrease after 10 or 12 variables for 2D and 3D problems. This phenomenon might be the reflect of a limitation of the method due to the combination of a large size genome and a population with few members favoring premature convergence.

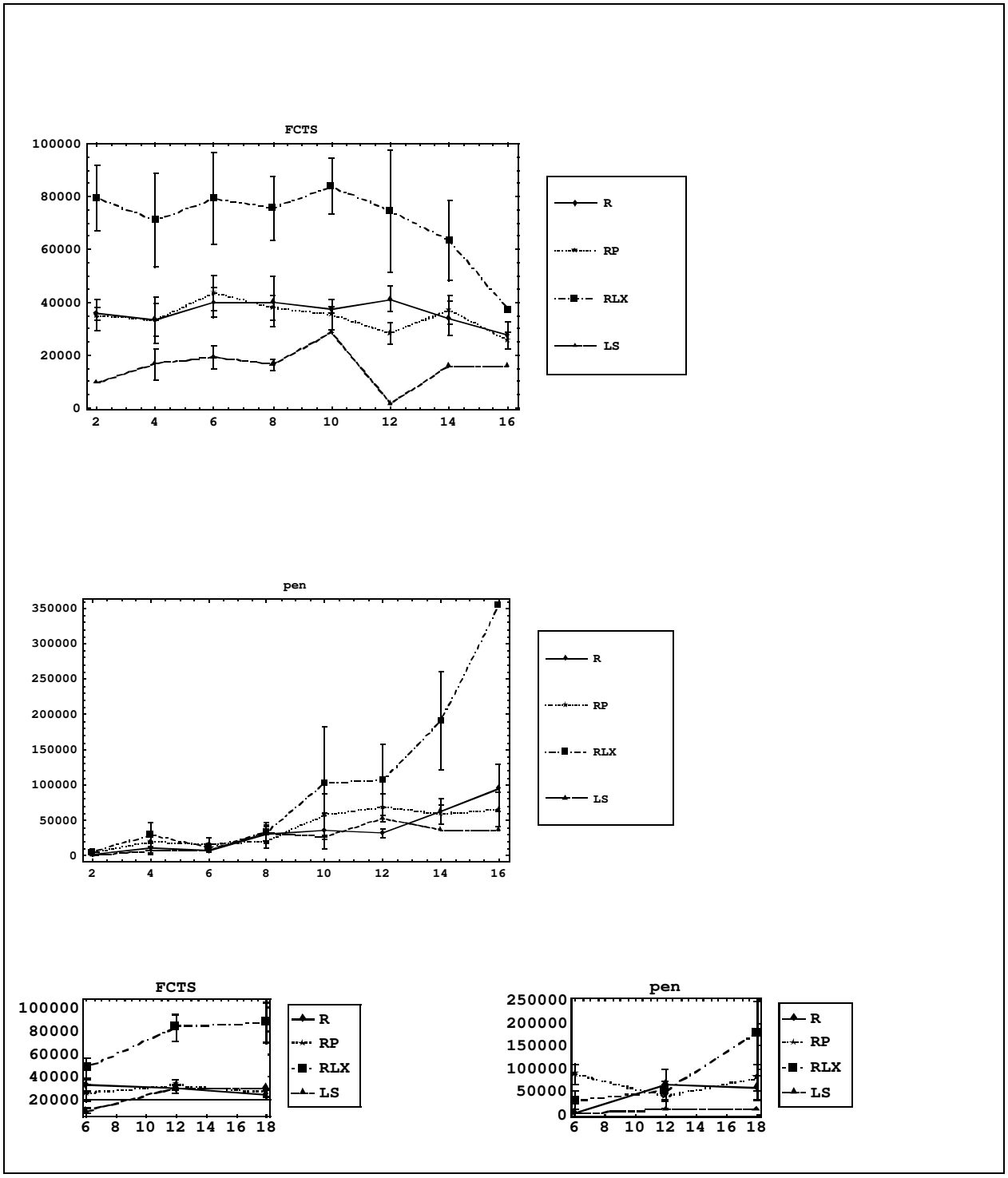

Figure 82. Evolution of the number of objective functions and penalty evaluations.

The speed of each method can now be compared by attributing a rank for each class of CDPs (Table XIX). This speed classification shows that the Relative Placement strategy is less expensive than its competitors, RLX coming next followed by the LS method.

Table XIX. Classification of the methods based on the number of objective functions and penalty function evaluation.

	R	RP	RLX	LS
FCTS 2D	3	2	3	1
FCTS 3D	2	3	4	1
PEN 2D	1	1	2	3*
PEN 3D	1	2	3	4*
Total	7	8	12	9

Note: '*' indicates that the rank was calculated by taking into account the additional cost of the local search.

All the methods presented are more expensive than the reference algorithm. although this difference remains small. This additional cost is usually the price to pay for obtaining better results.

Repeatability

The sizes of the error bars on the TEPs criteria prove that the methods can be re-run and still provide the same results. The fact that few error bars are overlapping indicates that the rank attributed to each method (Table XIX) will remain the same even if the number of test cases of each category increases.

This conclusion is also true for criterion C3 (Average distance from a reference plane) which indicates that the methods consistently converge toward the same front line. The larger size of the number of function evaluations error bars does not invalidate the classification given in Table XIX.

The spreading of the points inside the solution set is the only criterion for which all the methods seem equivalent and for which the validity of a clear classification seems doubtful.

In summary, it can be said that the extreme points and the front provided once by a method gives a fair approximation of its capability. The other criteria, (distribution of the

points and flatness of the front) present more instability. However this instability does not invalidate the classification of the methods.

Summary of the Academic Test Cases Results

The criteria used to investigate the process of Configuration Optimization were classified into two main categories. First the criteria regarding the solution set and second the criteria related to the methods.

Among the solution set criteria, the distances from the TEPs reflect accurately the relative complexity of the CDPs which might indicate that the complexity classification can be used to assess the performance of the CDOM on Single Objective Optimization problems.

Concerning the repeatability of the methods, the TEPs present the smallest error bars proving a stability of all the methods in the search for extreme objective values. All the amelioration proposed, in addition to providing better optima, exhibit a better stability than the base algorithm. This is not the case neither for the distribution criterion nor for the flatness for which there is no improvement between the stability of the reference algorithm and its derived versions.

The final classification, taking into account the TEPs criteria and the speed measures indicates that the Iterative Genetic Algorithm combined with a local search (LS) is the best candidate for the CDOM (Table XX). This algorithm is thus recommended whenever the additional computational cost of the LS routine is tolerable. For those who cannot afford this additional burden, the Iterative GA with Relaxation might be a better choice.

Table XX. Final Classification of the methods.

	R	**RP**	**RLX**	**LS**
TEPs	19	18	9	9
SPEED	7	8	12	9*
Total	26	26	21	18*

Note: '*' indicates that the rank was calculated by taking into account the additional cost of the local search.

The finality of this study is also to identify an eventual amelioration that might be brought to the CDOM. It was shown that the Local Search used for the LS algorithm introduced a bias in the feasible solution penalizing the maintainability objective function. It

is then recommended to change the LS routine by a search independent of any direction. It also was shown that using Relative Placement is not recommended when many objects can be permuted.

Although some methods perform better than others, it might be better to combine them for special applications. For example, the CDOM dedicated to mechanical assemblies might be a combination of the LS and RP or a combination of RP and RLX. Configuration problems involving many similar components can be submitted to a LS/RLX algorithm.

Engineering Applications

Two engineering configurations problems are submitted to the Iterative Genetic Algorithm with Relative Placement and Range Adaptation. Both systems consists of one or several static components, and multiple moving components that are placed in 3D space with respect to either the static objects or with respect to moving components. The Pareto sets of the 3 objective CDPs found by the method are presented as well as some corresponding configurations.

Aerospace Components

Systems destined to the aerospace industry, such as satellites and airplanes components must be thought of in order to respect the three objectives chosen in this work: compactness, location of the center of gravity, and accessibility of the components. In the following the CDOM is used to identify alternatives to the satellite CDP.

Problem Statement

A satellite, made of a base frame, an external hull, and internal electronic and mechanical components (Figure 83), must be designed. All the components must fit inside the hull and the inertia of the whole system must be minimum for energy savings and stability considerations. In addition to this first objective the Center of Gravity of the satellite must be put as close as possible to a target point located along the axis of the base frame, and the accessibility of the components must be maximized in order to ease construction and maintenance during an eventual repair intervention while on orbit.

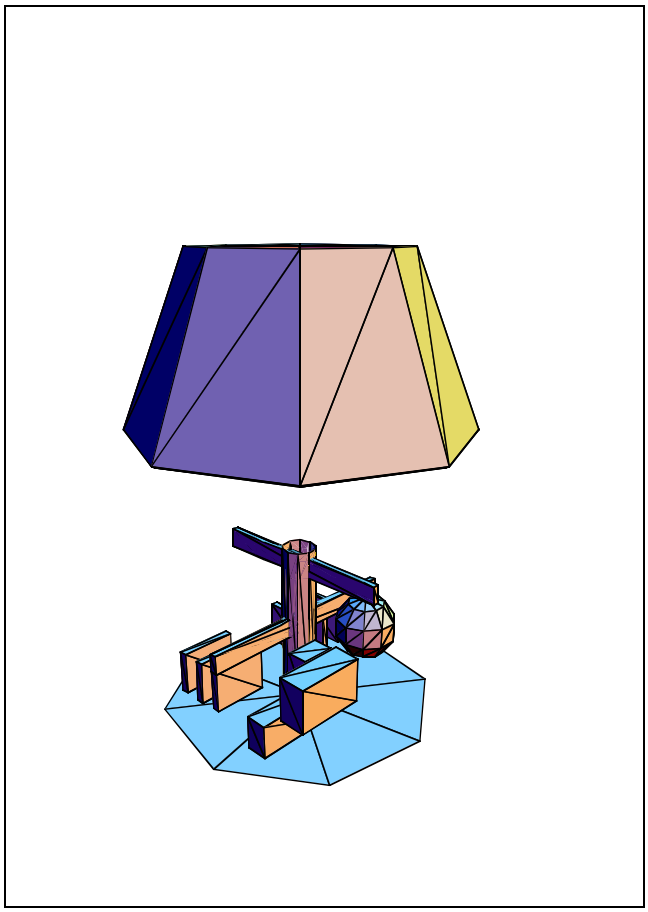

Figure 83. The satellite configuration design problem.

While compactness and center of gravity are implemented in the same way as for the test cases, accessibility is redefined for this particular application (Figure 84). The components can be accessed by two opposite paths along the diameter of the satellite. Hence, the accessibility objective is simulated by an additional component which is not taken into account in the calculus of the other objective functions and which simulates the volume necessary for a human being to access the components. The accessibility value is the number of components intersecting the access volume.

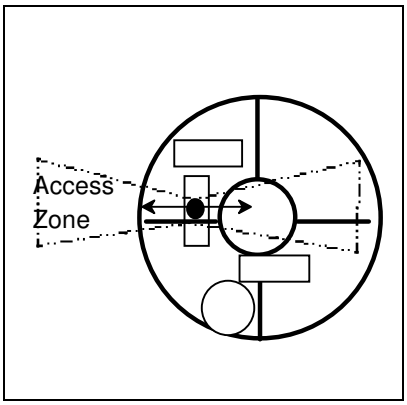

Figure 84. Accessibility of the satellite components.

The particularity of satellite design with respect to many other engineering design is the lack of history. This lack of data is due to the relative youth of the discipline and above all to the fact that no two satellites are alike. Hence, it is not surprising to see aerospace engineers designing satellites from scratch. In this context, a method such as CDOM might well be a strong support for stimulating conceptual design.

Objects and Variables

The satellite is made of seven principal parts that were identified to have a major impact on the volume, center of gravity and maintainability (Figure 85).

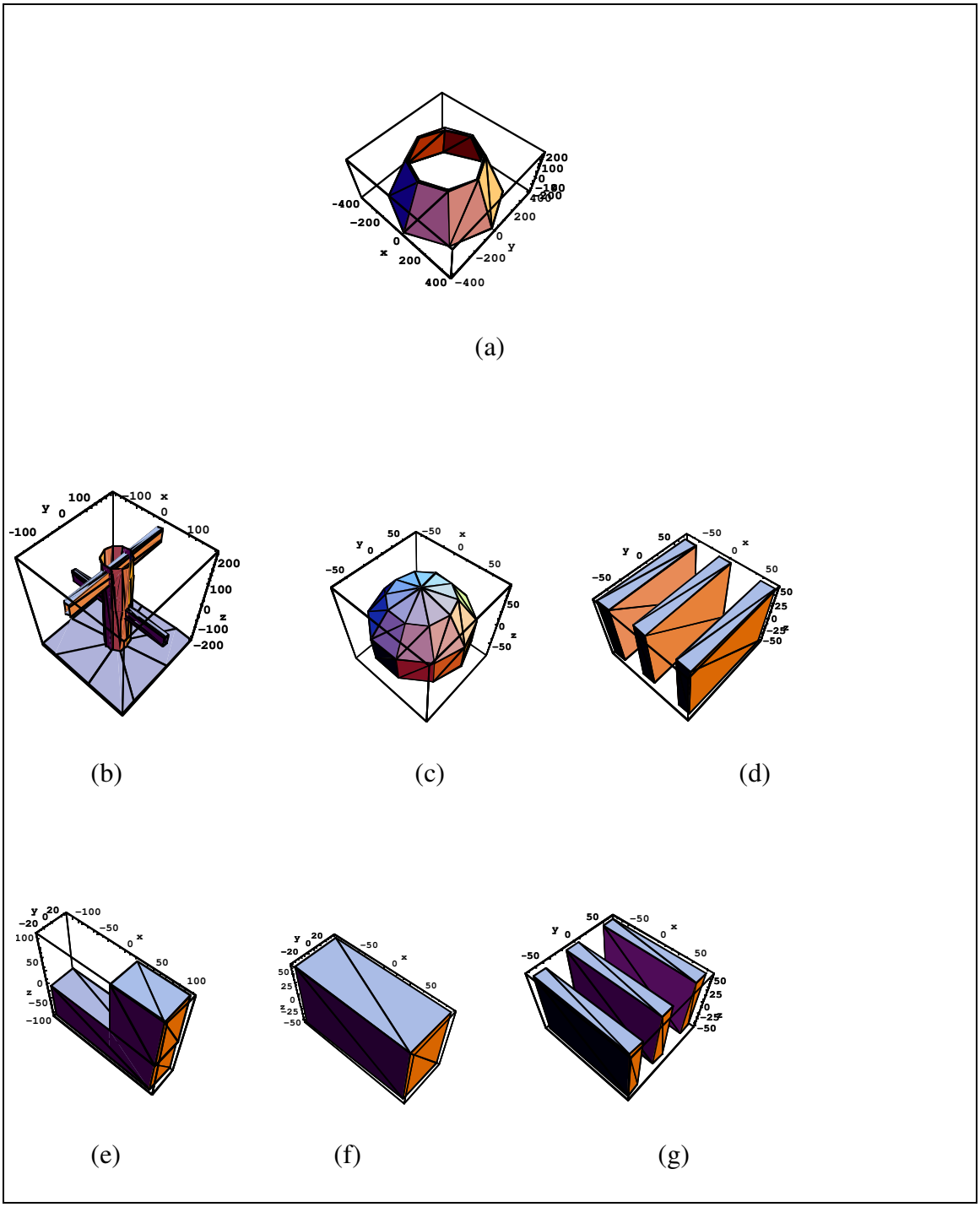

Figure 85. The satellite CDP components. From top to bottom and from right to left: the external hull (a), the base frame(b), a tank (c), sets of 3 electronic boards (d and g), some physical measuring equipment (e and f).

The base frame (b) and external hull (a) are both fixed with respect to the general coordinate system. The tank (c) and the first physical measurement equipment (e) can move in all directions inside the hull while the second physical equipment (f) can only move with respect to the first one along the y and z axes. One of the set of electronic boards can only move with respect to the first one (Table XXI).

Table XXI. Satellite CDP components degrees of freedom.

Component	Degrees of Freedom	Reference Component
1 (hull)	{0, 0, 0}	Reference
2 (frame)	{0, 0, 0}	1
3 (tank)	{tx, ty, tz}	1
4 (electronic)	{tx, ty, 0}	1
5 (measuring eq.)	{tx, ty, tz}	1
6 (measuring eq.)	{0, ty, tz}	5
7 (electronic)	{0, ty, 0}	4

Complexity Measures

This CDP involves 7 objects among which 2 are static, 2 are placed with respect to the absolute coordinate system, and 3 are placed relatively to another component. The number of variables of this ECDP is 11. Each variable can take 1000 different values between its lower and upper bound (these bounds are specified in the input files provided in Appendix B). The system volume is approximately the volume of the external hull (a).

These parameters as well as the shape definition of each object allow the calculation of some of the complexity criteria:

$Rg > 4.8$.

$Dil_{fa} > 0.09$

The dilution of the solutions is more difficult to assess. A Monte Carlo simulation is used to give an approximation of the solution dilution of each objective function. Only feasible solutions are retained. Assuming a Gaussian distribution of the values, the simulation provides the average and standard deviation of this distribution. The optimum

values obtained for each SOP provides the targets to achieve. The distribution and the target provide an approximation of the solution size in variables space.

$Dil_{sol1} > -23$

$Dil_{sol2} > -14$

$Dil_{sol3} > -13$

It must be noted that the Dilution of the feasible area indicates that the search will have difficulty to find feasible designs and consequently result in few points in the solution set. The dilutions of the solutions indicate that the compactness objective is more difficult to achieve than center of gravity and accessibility.

Results and Discussion

The best values of each objective function are reported in Table XXII. The best Compactness and Center of Gravity are obtained by submitting an Single Objective Optimization Problem to the CDOM. The best value of Accessibility is zero.

Table XXII. Results of the Single Objective Optimization Problems for the satellite CDP.

Objective	Solution	Speed	Penalty
Compactness	27	5950	4230
Cog	0		
Accessibility	0	2387	935

The configurations corresponding to the best compactness and best accessibility are displayed on Figure 86 and 87.

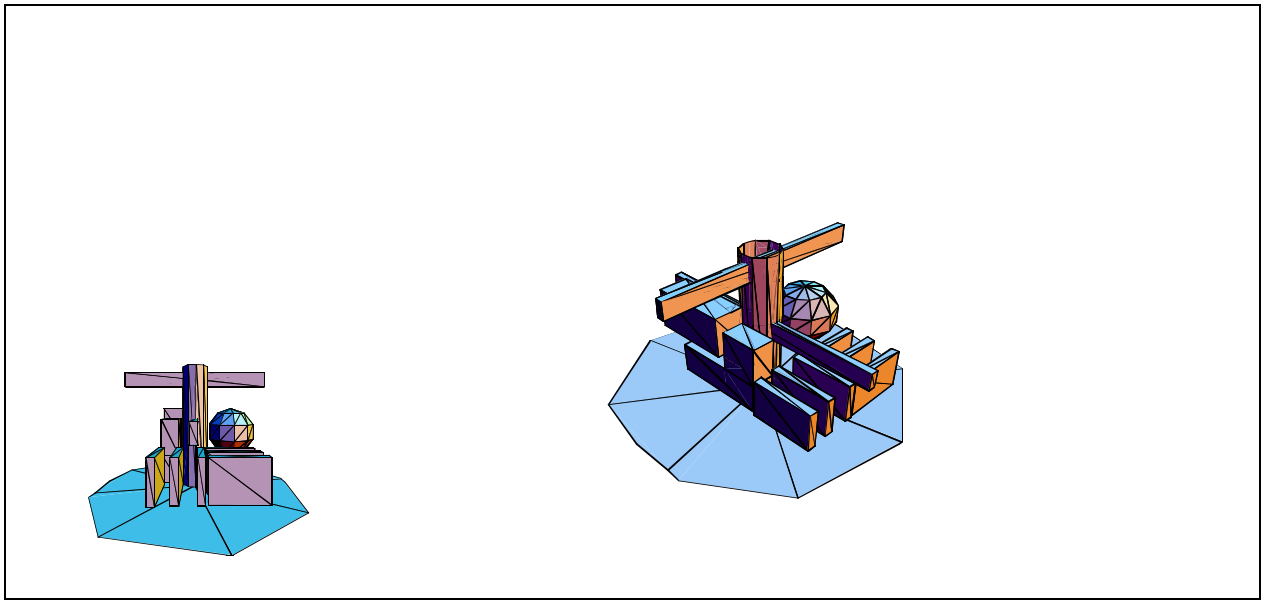

Figure 86. Example of maximum compactness configuration.

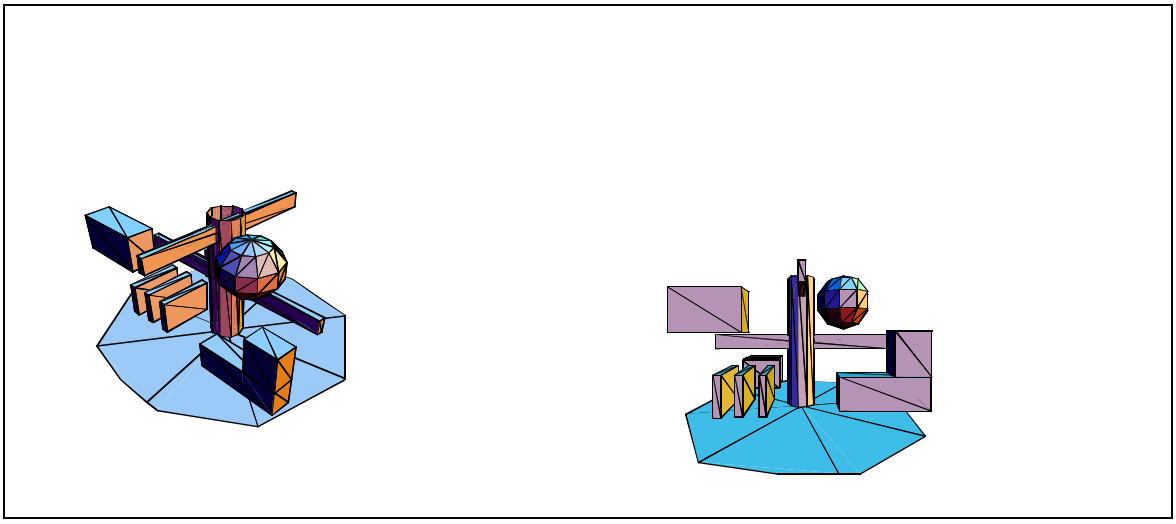

Figure 87. Example of maximum accessibility configuration.

The multicriteria version of the satellite ECDP is submitted to the RLX method. Its results are summarized in Table XXIII. As expected the number of points discovered by the CDOM is low (Figure 88). It was conjectured that the low feasible area is the cause of this problem. An additional clue corroborating this hypothesis is the high number of penalty function evaluations compared to the number of function evaluation Table XXIII.

Table XXIII. Results of the multi objective optimization problem for the satellite CDP.

Objective	Extreme Solutions	Functions	Penalties
Compactness	28.1	23920	148945
Cog	0.92	same	same
Accessibility	0	same	same

However, the cloud of points seems relatively evenly distributed. This was already observed on small solution sets in the test cases. It seems that the cloud is also well spread along the compactness and accessibility axes, providing a range of compactness between 28.1 and 29.5 and an accessibility between 0 and 4. However, a wider range for the center of gravity was expected: it varies between 2.3 mm and 23 mm (knowing that the satellite is 1 meter in diameter). Finally, the large distance between the best possible compactness and the extreme point found in the multi criteria problem along the same axis was predicted by the analysis on the cube CDPs.

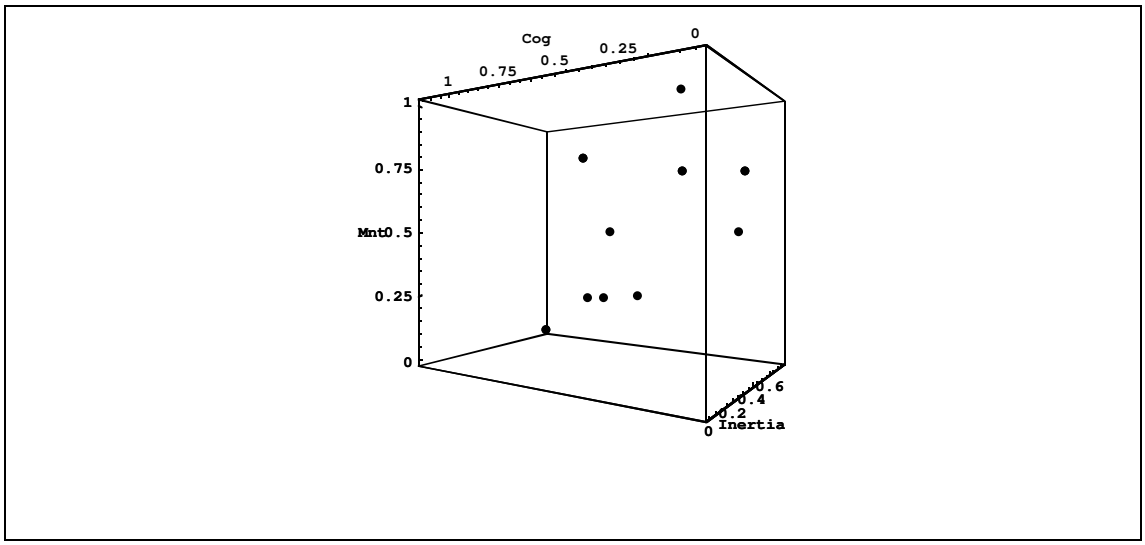

Figure 88. Three - objective Pareto set provided by the CDOM.

From the mechanical engineering point of view the results seem correct i.e. as the configurations become less maintainable. it is possible to obtain a smaller volume and a

better center of gravity (Figure 89). The small range of the center of gravity might be due to the relative high mass of the frame (Appendix B, satellite input file, component sat2).

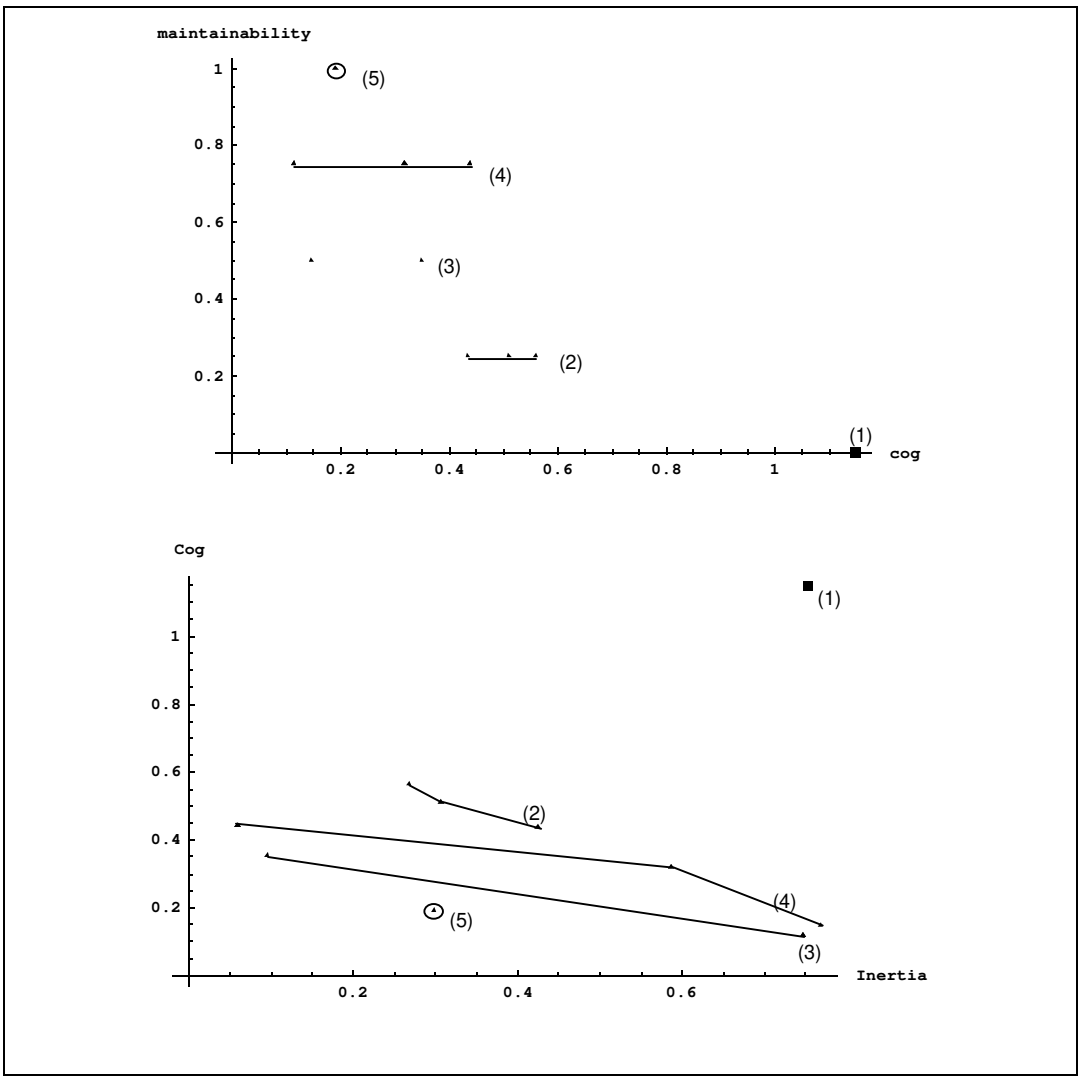

Figure 89. 2D views of the three objective Pareto set provided by the CDOM. All the values were normalized with respect to their interval of variations (inertia log: [28.1, 29.5], center of gravity (cm) [0.92, 9.18], accessibility (number of objects in the access path) [0, 4]).

There is no guarantee that the non-convex shape of the Pareto set provided by the method (curve 4) is the reflection of the Pareto set solution of the MOP. This shape might translate the inability of the CDOM to refine this particular boundary.

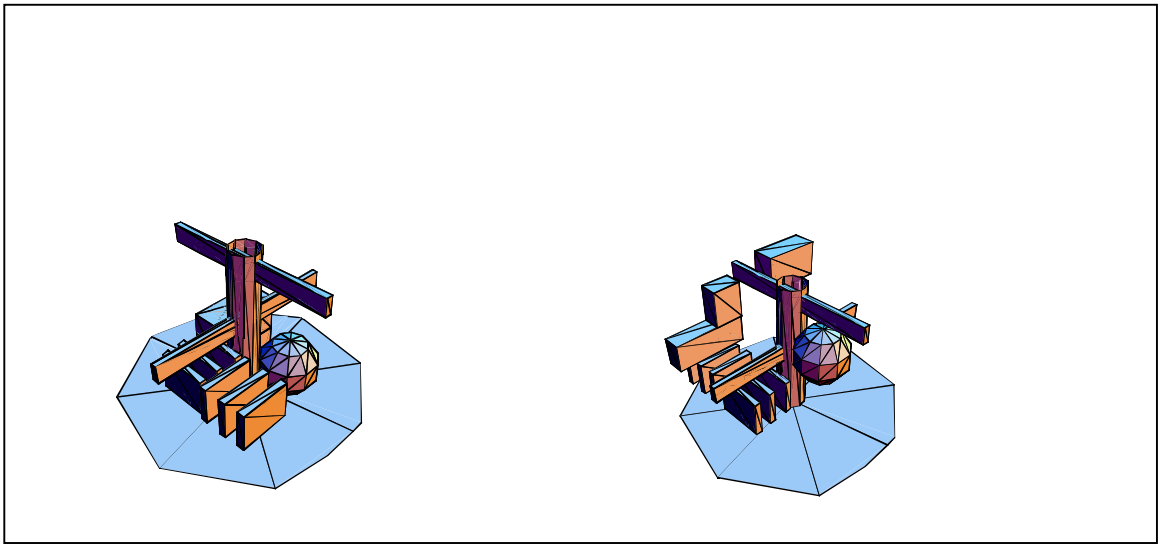

Figure 90. Two extreme configurations. At left, the maximum compactness configuration discovered by the CDOM. At right, the maximum maintainability configuration.

The output provided by the CDOM provides useful information of several kinds for the designer. First, a set of alternate optimal solutions is given (Figures 89 and 90) guiding the positioning of the components. Second the Pareto set provided can be used as a basis to compare 'hand generated' solutions. Third, mechanical characteristics of the system might emerge from the study of the solution set. For example the fact that the set is small on the center of gravity axis might prove that the moving components have little influence on this objective (which is a good point for this case).

Car Engine

For mechanical systems such as a car engine the center of gravity must be as low as possible while the components must be able to be accessed as easily as possible. This later goal is contradictory with a minimum volume engine. Contrarily to the satellite ECDP, the car engine problem benefits from a long history and many data that can be used for comparison. Although creating new design by patching old ones is harmful, it was showed that this is a common way of designing, especially in the car industry (National Research

Council 1991, Womack 1991). This is why, using the CDOM to generate new alternate design is interesting.

Problem Statement

Several components identified as having an impact on the car engine size, weight and maintainability must be placed such that the 3 objectives chosen in this work are optimized at the same time. Some other components, such as the clutch, the starter, the transmission axle, are purposefully removed from the simulation since they are static with respect to the motor block (Figure 91).

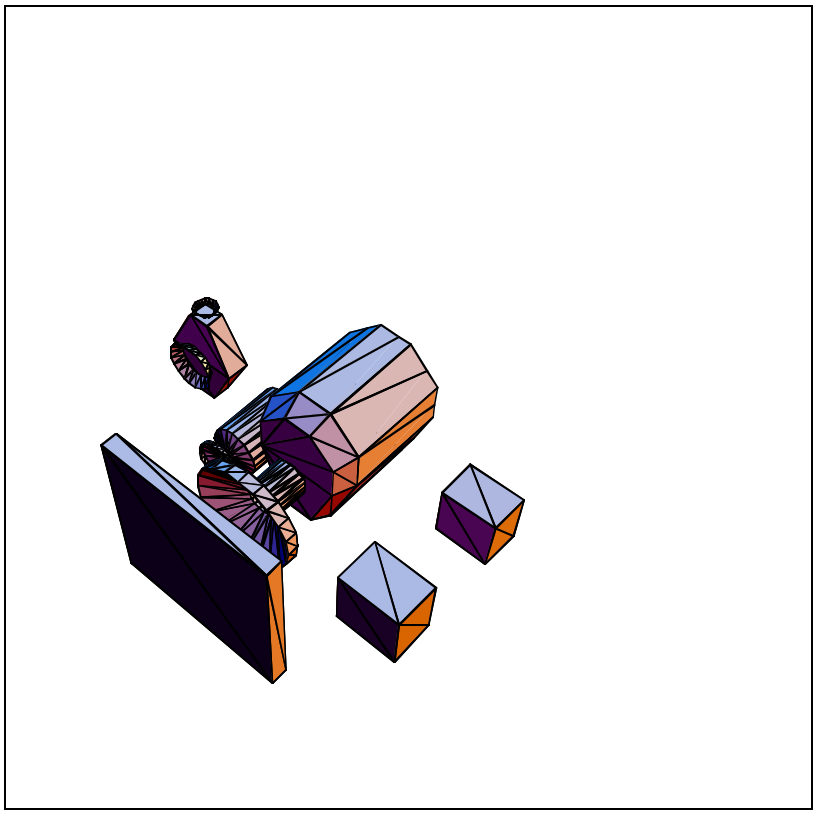

Figure 91. A simplified car engine submitted to the CDOM.

Is this case accessibility is defined as it was for satellites, i.e. a static component simulates the volume necessary to access a given element. Then the value of the accessibility is calculated as the number of components intersecting this volume (Figure 61).

Objects and Variables

Seven components are selected for placement under the hood of a car (Figure 92).

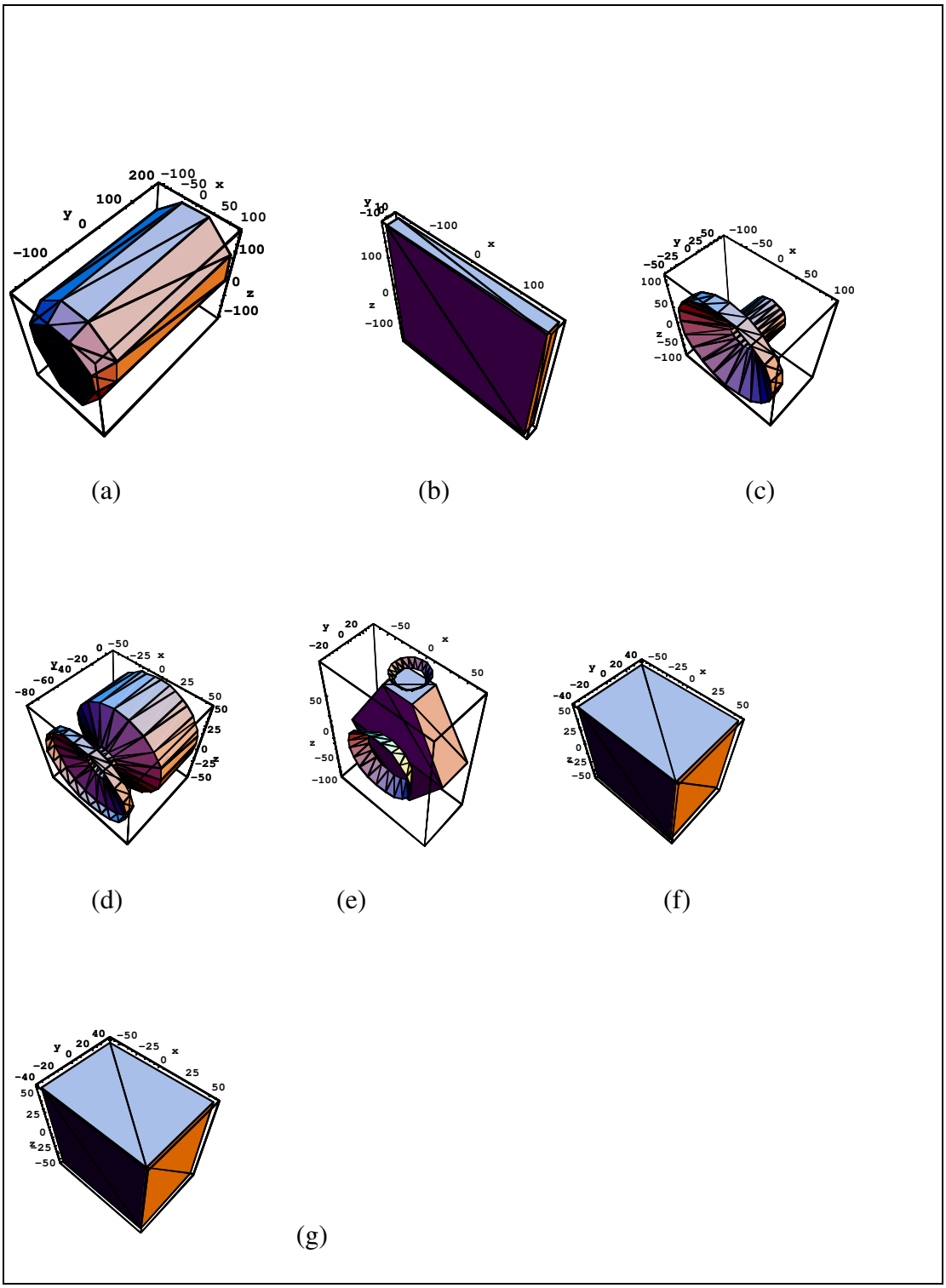

Figure 92. The car engine CDP components. The motor block (a), the radiator (b), and its fan (c), an alternator(d), a power steering tank and pump (e), a battery (f) and a water tank (h).

The motor block is used as reference for all the static components and for the axis of the car. It is purposefully put on the longitudinal axis (x-axis). The y-axis is horizontal and perpendicular to the longitudinal axis and the z-axis is vertical. The fan (c) can move along the x and z axes with respect to the motor block. The ranges chosen for its translations constrain its position to be in front the motor block. This constitutes the first example of mechanical constraint satisfaction using relative placement. If absolute placement had been chosen, an additional inequality constraint ($y_{fan} > y_{motor}$) would have been necessary. The radiator (b) is placed with respect to the fan (c). Its only degree of freedom allows a transversal displacement. The alternator (d) and the power steering pump (e) are placed relatively to the motor block such that the pulleys axes can always be parallel to the exit pulleys of the engine. They also must remain in the same plane. These constraints are additional mechanical constraints that would have complexified the variable space if absolute variables had be used. Finally, the battery (f) and water tank (g) can be put anywhere. The degrees of freedom of each component are summarized in Table XXIV.

Table XXIV. Car engine CDP components degrees of freedom.

Component	Degrees of Freedom	Reference Component
1 (motor block)	{0, 0, 0}	Reference
2 (radiator)	{tx, 0, 0}	3
3 (fan)	{tx, 0, tz}	1
4 (radiator)	{tx, 0, tz}	1
5 (power steering)	{tx, 0, tz}	1
6 (battery)	{tx, ty, tz}	1
7 (water tank)	{tx, ty, tz}	1

Complexity Measures and Classification

As in the satellite CDP, the car engine problem involves 7 objects. The motor block is static and placed at the origin of the coordinate system, 4 objects are effectively placed with respect to another component, and 2 can move without restrictions. The number of variables of this ECDP is 13. Each variable can take 1000 different values between its lower and upper bound (these bounds are also specified in the input files provided in Appendix B). The system volume is approximately 800x600x400 (the approximate size of each component can be seen on Figure 91).

The complexity criteria for this case have the following values:

$Rg > 5.3$.

$Dil_{fa} > 0.98$

$Dil_{sol1} > -19$

$Dil_{sol2} > -14$

$Dil_{sol3} > -11$

It must be noted that contrary to the satellite CDP, the Dilution of the feasible area indicates that the search will easily find feasible designs and consequently provide many points in the solution set. The Roughness of the landscape remains inside acceptable bounds (see complexity classification Table XVII and Cube CDP results).

Results and Discussion

The same procedure as the one used for solving the SOPs on the satellite CDP is used for this real case. The best values of each objective function are reported in Table XXV.

Table XXV. Results of the single objective optimization problems for the car engine CDP.

Objective	Solution	Speed	Penalty
Compactness	41.15	6561	4716
Cog	17	5398	1022
Accessibility	3	3911	967

The configurations corresponding to the best compactness and best accessibility are displayed on Figures 93 and 94. In the best compactness configuration, the power steering pump and the alternator come slightly under the engine block in order to niche on both sides of the V shape. The battery and the water tank tend to have the same behavior but have too much height and thus cannot slide under the motor. The fan and the radiator are also slightly shifted on the right in order to counter balance the effect of the alternator and battery on the left.

In the maximum accessibility solution all the components are placed as far as possible from each others and as high as possible, which is the exact opposite of what is happening for the compactness objective.

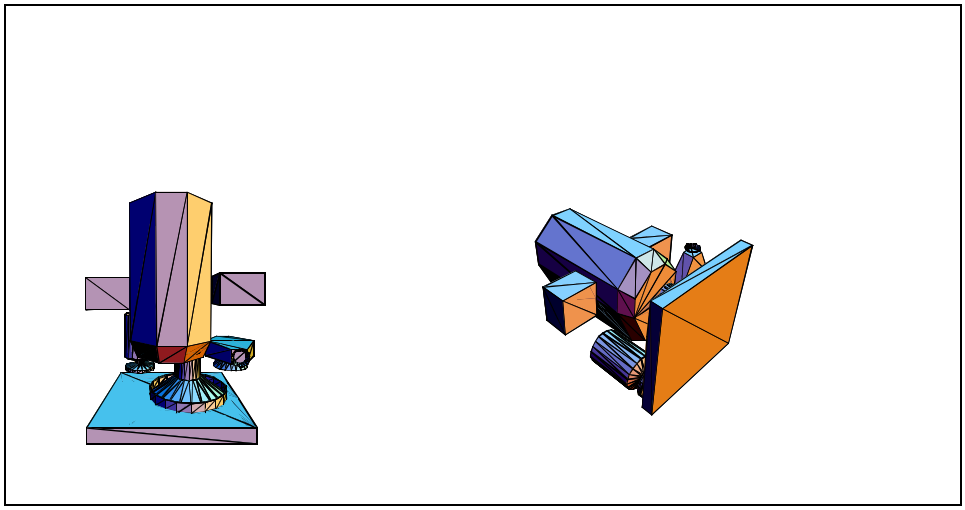

Figure 93. Example of maximum compactness.

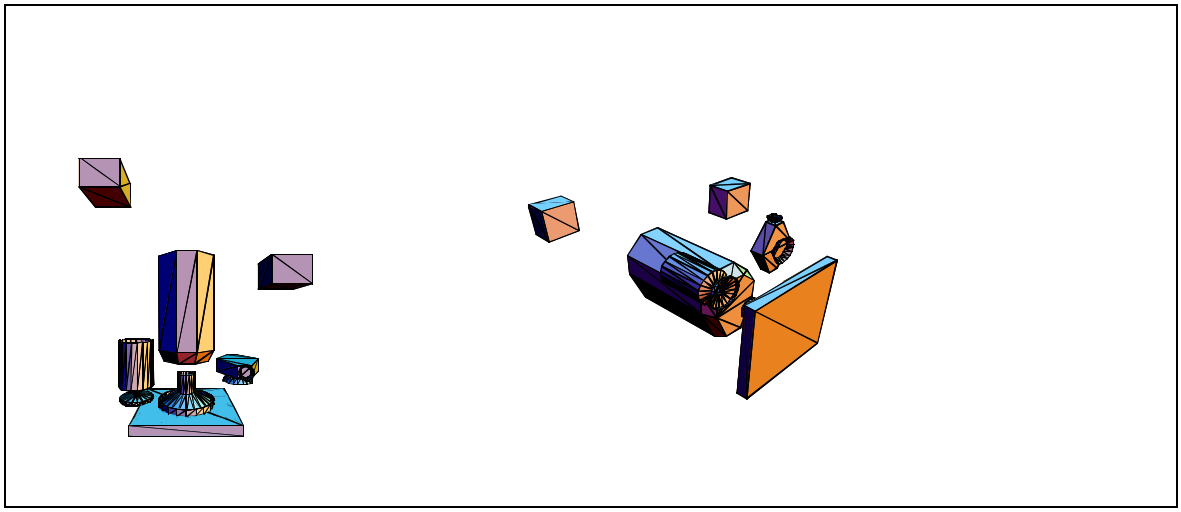

Figure 94. Example of maximum accessibility.

The RLX method is also used in this case for solving the multicriteria version of the car engine ECDP. The number of solutions found by the CDOM reaches its maximum (Figure 95). This was predictable thanks to the dilution of the feasible area. However, this factor does not constitute a proof that the solution set is large. The only guaranty given concerns the relative ease with which the CDOM is able to investigate the design space. The true reason for the discovery of many points is the size of the Pareto set. It was shown with the single optimization performed that the car engine presents extreme solutions that are, in variable space, very different and thus increase the chance of having a large Pareto set.

As was expected, the number of penalty function evaluation was low due to the large feasible area (Table XXVI).

Table XXVI. Results of the multi objective optimization problem for the car engine CDP.

Objective	Extreme Solutions	Functions	Penalties
Compactness	42.5	48850	9563
Cog	21	same	same
Accessibility	3	same	same

The extreme compactness values oscillates between 42.5 and 43.5 (i.e. the solution gives the choice between solutions that can be more than twice less compact). The center of gravity is at a minimum distance of 10 cm and at a maximum of 17.3 cm away from the target. The accessibility exhibits the best range with a minimum of 3 and a maximum of 6. The minimum value cannot be less than 3 due to the fan which must remain in between the radiator and the motor block.

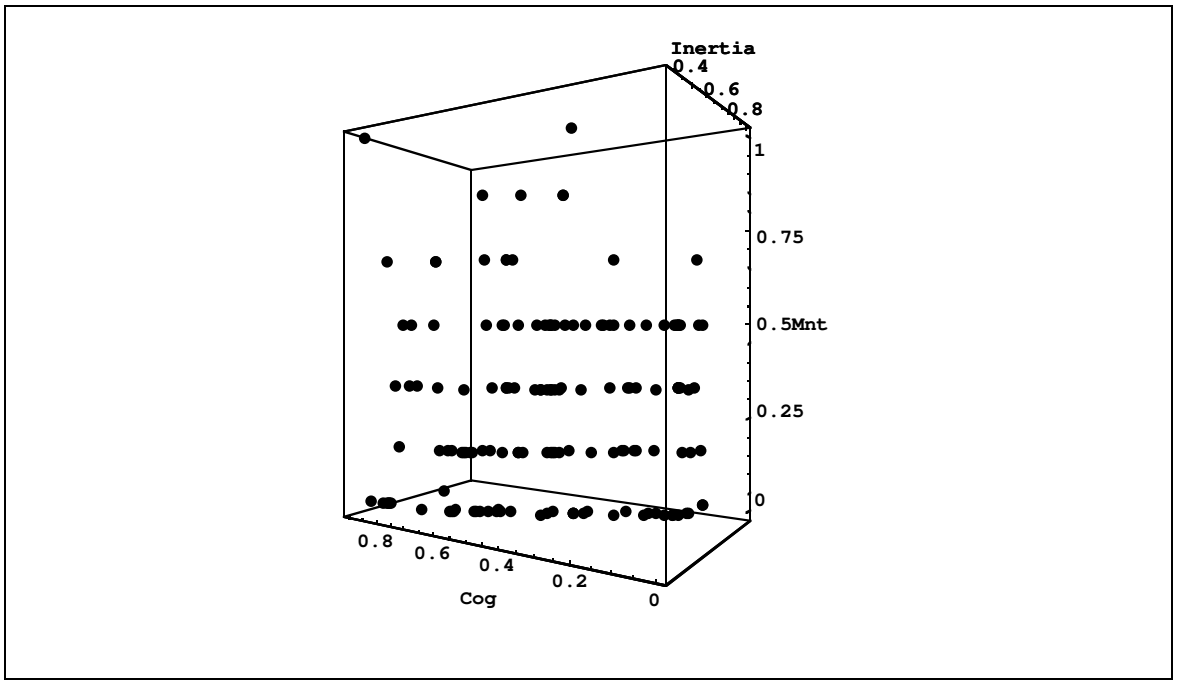

Figure 95. Three objective Pareto set of the car engine ECDP provided by the CDOM.

Finally, it can be noticed that the CDOM provides better center of gravity and removability than compactness configurations. This was correctly predicted by the Dilution of the solutions. Consequently, the final solution exhibits solutions with fair compactness and good maintainability (Figure 96-b), with however a better compactness achieved at the same time as bad accessibility (Figure 96-f).

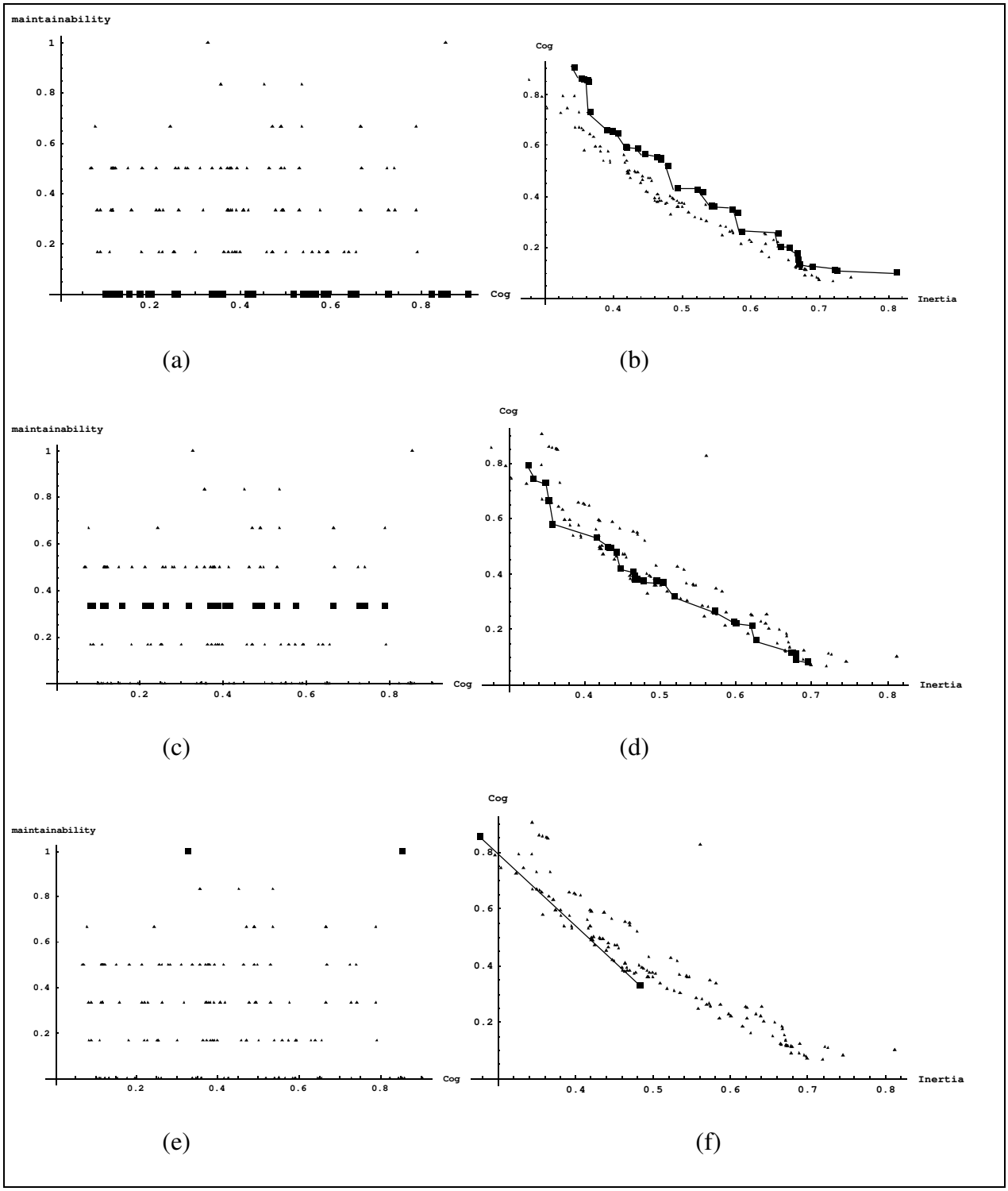

Figure 96. CDOM solutions. Three different layers of the solution set provided by the CDOM on the car engine ECDP. The top two figures, (a) and (b), represent the best Removability. The middle figures, (c) and (d) represent the Pareto set corresponding to a medium Removability. The bottom two figures, (e) and (f), represent the Pareto set of the configurations of worse Removability.

From the mechanical engineering point of view, the extreme solutions (Figure 97) can be compared to existing designs. First, the solutions presented do not align the fan and the longitudinal axis of the motor block. Since no fluid flow factors and cooling was taken into account, the fan and the radiator are moved in order to optimize inertia and center of gravity objectives. In most cars the fan is aligned with the central longitudinal axis of the car. Putting the battery and water tank close to the motor might be a bad choice, first because the coolant must be put away from heat and second because the battery may take the place of another component necessary to the motor such as the air filter. The good point of this simulation concerns the power steering pump and the alternator, which exhibit fair behaviors.

These results prove that additional static components must be taken into account. This might be possible since the Roughness calculated for this problem is low. Additional relative placement constraints are necessary for the battery and the water tank. This does not cause an increase in complexity of the problem.

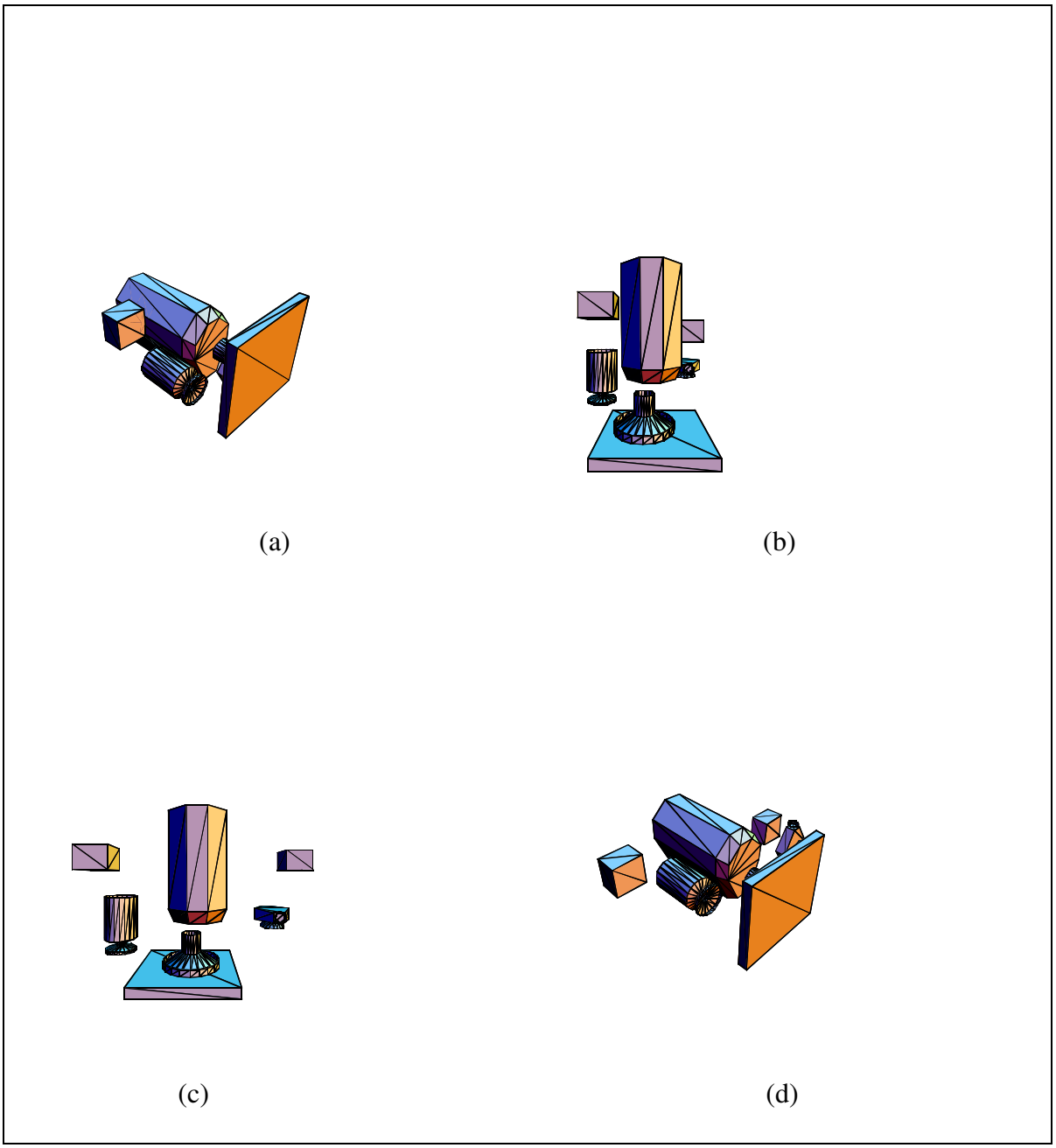

Figure 97. Maximum compactness configuration (a and b) and maximum accessibility (c and d).

Conclusion

In this Chapter, the performance of 4 methods were compared for solving several CDPs involving the three criteria: compactness, center of gravity and maintainability. One of them, the Iterative Genetic Algorithm with Range Restriction, R) was taken as reference

since already proven to address successfully the search for multiple solutions in multicriteria optimization problems (Grignon and Fadel 1997). The remaining three methods brought a change to the R strategy, accordingly to the analysis done in Chapter III, in order to improve its performance. Two methods reached this goal while a third one only provided a small advantage. The LS strategy (R modified by a local search applied to each unfeasible solution) performed better than its two competitors with however a highest computational cost. Coming next, the RLX strategy (R enhanced by an adaptation of the variable ranges in function of the success of the search) is less expensive. Finally, the RP strategy (R with relative placement of the objects) can be considered as equivalent to the reference.

Addressing multicriteria CDPs meant that the Pareto set was unknown, hence several quality criteria had to be determined for rating the quality of the solution obtained with respect to the known data. Among them, the distance from the extreme points (solutions of the three single objective CDPs, one for each objective function), the number of points, and the average distance from the reference plane were the most reliable measures for assessing the quality of the solution. The flatness criterion was deceiving since it depicted a bad flatness for clouds of points not parallel to the reference plane. The criteria measuring the speed of the method were accurate. Keeping separate the number of penalty function and the number of objective function evaluations allowed to draw an interesting conclusion about the LS search. It was indeed conjectured that a local search brings more feasible solutions in the population of the Genetic Algorithm and thus the latter can evolve more quickly toward feasible design. The lower number of penalty functions evaluation exhibited by LS corroborated the hypothesis. Finally the drop in the number of function evaluations when the number of variables is larger than 12 tend to indicate a limitation of the method. This drop is attributed to premature convergence of the Genetic Algorithm due a small population size.

The test cases results presented a close match with the complexity classification proposed in Chapter III. This good result indicates that the approximations done for evaluating the Dilution of the solutions through Monte Carlo simulations, the Dilution of the feasible areas, and the Roughness are valid. However, it must be emphasized that the size of the Pareto set remains unknown and no conjecture can be done about it except very crude lower and upper bounds. The fact that the complexity of a CDP can be assessed using simple

measurable characteristics (size of the system, size and number of the components as well as the type of placement), is encouraging since this method is dedicated to engineers.

The generality of the CDOM is assessed by its robustness in front of the different shapes and component numbers involved in the test cases. The decrease in performance is, on average, linear, while the complexity increases exponentially (the complexity criteria are all expressed as logarithms). However, three limitations must be taken into account. First, the number of penalty functions increases exponentially. Second, the size of the genomes increases linearly in function of the number of variables times the number of points in each cloud of the GA population. For a population composed of 15 point clouds, the genome size of each child is 15*(number of variables)*(number of bits per variables). Third, relative placement can only solve simple mechanical functional constraints (parallel, perpendicular, concentric, fix) hence, each component can only be placed with respect to a single reference.

Eventually, the analysis of the test cases pointed two technical flaws in the LS and RP strategies. First, the LS introduced a bias in the feasible configurations penalizing the maintainability objective function. Second, it is preferable to avoid the RP strategy when many components can be permuted.

CHAPTER VI

DISCUSSION AND CONCLUSION

Introduction

This research, started nearly four years ago at Clemson University Mechanical Engineering Department, and ends at the Department of Research and New Technologies of Dassault Systemes (France), one of the leading CAD/CAM company in the world. What began with the optimization of the center of gravity positions of rectangles for simulating the payload of a truck was extended to 3D space, freeform objects and multiple criteria, and promoted this research in one of the most challenging research topic currently under investigation in the Computer Aided Design industry : the design of full optimized assemblies.

For the first time, the computational power, the data infrastructures and mathematical methods are available to begin to address this area of the Digitized Mechanical Engineering Design Process. If this research area achieves its final goal, the consequences on everyday life might be dramatic for all the industrial sectors. Cars, trucks and planes consuming less, buildings being able to endure the most severs hearth quakes, products costing less with enhanced safety, increased comfort are among the many advantages that full assembly optimization might bring.

Overview of this Dissertation and Achievements

Today's design process practice shows that companies are heavily relying on CAD/CAM software for creating parts and assemblies. During this process, the automatic tuning of the product parameters (dimensions, shapes, materials) has been, until recently, the exclusive domain of the part design stage. This process is now spreading to the conceptual stage and to the sub-assembly and assembly design of the product. However, the fact that part optimization is less complex, better known and solved more efficiently by specialized methods, prohibited the change of part parameters due to assembly type constraints. It is not rare, in today's applications, to consider part and sub-assemblies as constant whenever system level optimization must be done. This decision simplifies the optimization problem

by removing the possibilities of change in the final product. As the engineers keep building the system, constraints appear, reflecting different knowledge fields, that link parameters at various levels and thus break the well established and safe feeling of the hierarchical organization of the full system. At the present moment no satisfactory solution is implemented to optimize these additional relations.

The problem addressed in this work can be summarized as follows: "how can assembly level optimization be performed on industrial projects ?" The main goal is to define a method able to address this problem. This implied, in order to build a method able to tackle realistic engineering problems, that free-form components could be taken into account as well as the functional constraints that hold all the system components together. In addition to these first requirements the necessity to address multiple goals at the same time was also dictated by realistic engineering design considerations. Translated in mathematical terms, configuration problems can be reformulated as the search for optimal solutions of constrained non-linear multi-criteria optimization problems. The review of the industrial applications showed that no assumptions could be made on the type of components and systems. However, system optimization usually occurred after components shapes and their functional links were settled. From the mathematical representation point of view, the variables (displacements) can be considered as continuous in all cases.

Once this was assessed three areas were identified to be necessary for addressing the goal:

- define the complexity of an Engineering Configuration Design Problem (ECDP),
- define the method (called CDOM),
- measure its performance.

In order to address the first goal, a mathematical formulation was proposed that, unlike many formulations found in the literature review, could address any type of components (involving non convex, hollow, sharp edges components). This formulation relied on the use of continuous variables for defining the allowed movements of the components and on penalty functions for penalizing unfeasible configurations. Then, based on the notion of function landscape, four criteria were chosen to answer the first need i.e. evaluating the complexity of the ECDPs. Trend, Roughness, Dilution of the feasible area and

Dilution of the solution were defined for any type of C^1 functions and linked to simple characteristics of the ECDPs (size and number of the components, size of the system).

The second goal, the definition of the method, was addressed as finding multiple global extrema of a non-linear optimization problem. A Genetic Algorithm working on population of sets instead of population of individual points was proposed for searching the variables space and for providing multiple solutions for three reasons. First, the GA works on several designs at the same time; second, it is able to deal with highly non-linear functions; and third, it capitalizes on the knowledge of its previous trials. The analysis of the complexity classification of some well-known packing problems provided clues for proposing three enhancements to this initial method. First, the penalty should always be combined with the objective function for unfeasible configurations evaluation, then, the use of a Local Search, and Relative Placement contributes to decrease the CDP complexity. Eventually, to increase the chances of the CDOM to escape local minima, an adaptive range strategy was suggested.

Before verifying the validity of these choices, some criteria had to be designed for measuring the quality of the Pareto set obtained using the CDOM. Three criteria were chosen for measuring the quality of the set itself (distance from the extreme planes, distribution of the points, flatness) and two for measuring the performance of the CDOM (speed and repeatability).

A series of cubes CDPs and two engineering cases were submitted to the method for validating the hypotheses and the complexity classification. The conclusion of this part of the study pointed out that the Local Search and Range Adaptation enhancements were the most effective. Relative placement did not bring a clear advantage over the base strategy when dealing with CDPs in which the relative motion is not linked to a functional need. However, this bad performance vanishes as soon as mechanical assemblies are submitted. These modifications help to enhance the performance of the method proposed for achieving the main goal of this study.

Among the two types of criteria proposed for first rating the CDPs complexity and second for rating the CDOM, only the former were clearly satisfactory and can be reused in further studies. Several criteria are to be rethought for assessing the quality of a Pareto set.

Discussion of the Results and Criticism

Configuration Design Optimization Method

Even very simple instances of configuration problems are known to be intractable. Hence, trying to tackle the ECDPs with their thousands of variables and their hundreds of non-linear constraints seems out of reach of our possibilities. In this study it was proposed to bypass the difficulty by working on fully constrained assemblies and authorize some of their components to move by relaxing some constraints. Just like a human being is able to build a puzzle, in spite of the exponential growth of its different configurations, because he knows that colors and shapes must match, an engineer can make use of his other knowledge of the system to allow a restricted number of components to move "intelligently".

The use of a genetic algorithm population to discover several solutions to an optimization problem answers the fourth fundamental question. Three different approaches are possible. First, sequentially use a method that is able to find a different solution after each run. In the view of the author, this solution does not make use of the information gathered during previous runs and thus is less efficient in producing a multiple solutions to the MOP. Second, use a population based strategy such as a genetic algorithm. The review of the research done in this direction strengthen the conclusion that the primary design of the GA, i.e. clustering the population points around a single value, is contradictory with the goal of this work which is spreading the population along the Pareto set. This remark gave birth to multiple efforts done by researchers in this domain. Third, a new formulation, proposed in this dissertation, brings back the use of the genetic algorithm to its original design which seems more adapted than any algorithm proposed in this field. The information gathered for finding a Pareto point is implicitly reused for discovering neighbor Pareto points. However, many questions remain concerning the use of genetic algorithms for solving multi objective optimization problems: first, questions concerning the relative influence of the non-inferior point definition (characterizing Pareto points) with respect to the set spreading in the objective function design; second questions concerning the preference for wide sets and the conservation of extreme solutions in the population. Eventually, the presence of multiple objectives in the CDP prevented the use of local methods on the objectives themselves (since the local search would have lead to three different points in variable space). The rank based (and thus discrete) objective function was an impediment for using local search on the final

objectives combination. Thus, the local search was restricted to the discovery of feasible configurations. The small size of the population might be the reason of the small number of variables that the CDOM is able to handle (premature convergence). This is partially counterbalanced by the iterative use of the GA.

The iterative use of a GA means that the knowledge must be capitalized from one run to the other otherwise a large part of the iterations' interest is lost. This is achieved, on one hand, by using the storage of the best solution taken as an ever-changing reference, and, on the other hand, through the variable ranges restrictions which narrow the GA investigation around the best solutions found so far. This restriction should go along with a reduction of the genome size in order to keep the final precision of the variables constant (and eventually speed up the convergence).

The variable range relaxation is an attempt to make the search less sensitive to premature convergence on local optima while keeping the control of the variables bounds. In GA research, this can typically be done through an increase of the probability of mutation with however one drawback: the difficulty to control the range in which the new design variables will be. The improvement brought by this simple strategy is encouraging and leaves open the possibility to further adapt the behavior of the method while bringing a beginning of an answer to the third fundamental question.

Complexity Classification

The complexity classification proposed for rating the CDPs provides an answer to the first fundamental question. It is also a complement between the combinatorial complexity relying on the size of the problem input and the ε-complexity used for continuous functions. It contributes to the identification of the configuration problems features that make them easy or difficult to solve.

However, the relative influence of the three criteria remains to be found. For example using equal scales on all the axes of the classification changes the relative complexity of some CDPs (compare Figure 70 and Figure 98).

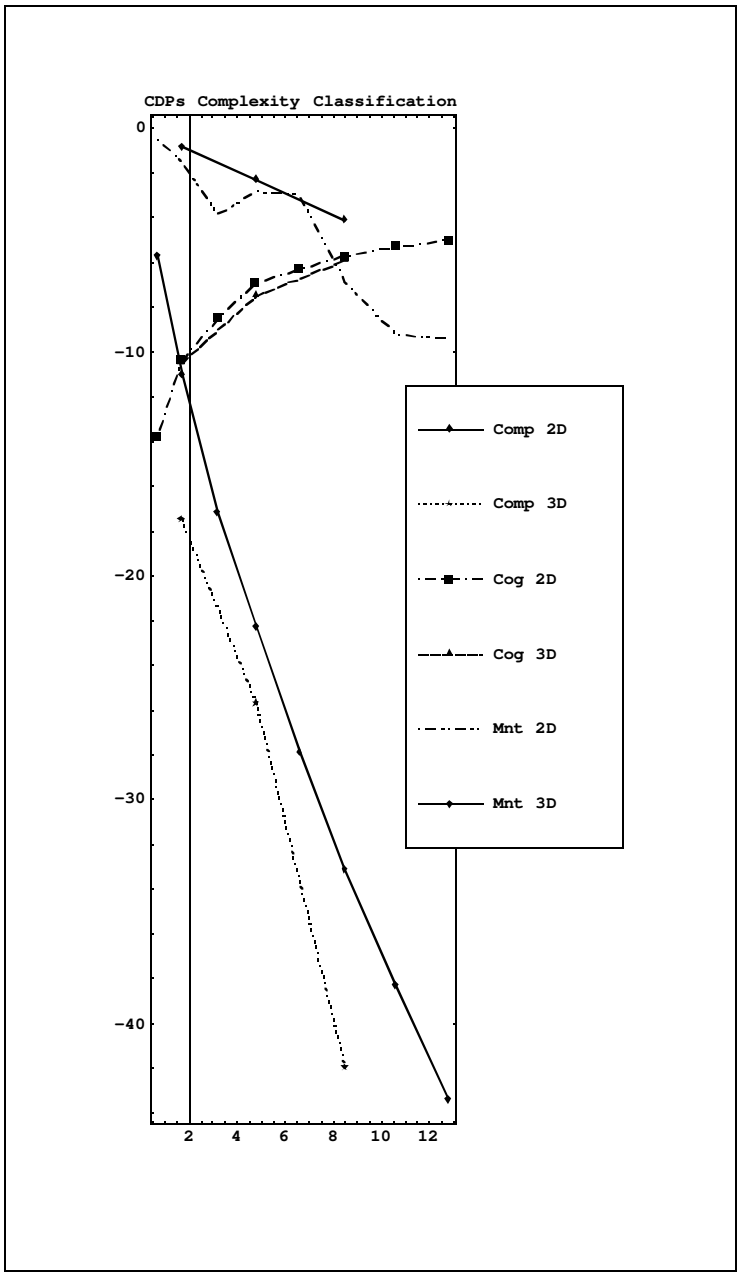

Figure 98. Equal scale cut of the complexity classification for Dil_{fa} = 1/3. The vertical axis holds the logarithm of the solution dilution and the horizontal axis corresponds to the logarithm of the roughness.

The elimination of the Trend from the selected criteria used to define this classification seemed to have little influence on its validity. This might be due to the redundancies between the Trend and the other three criteria. This factor is however

considered as the most significant since being an integral measure of the clues available in the landscape for finding the global optimum.

Pareto Set Quality Criteria

It was difficult to provide general criteria able to define how a "good" Pareto set looks like when this same set is unknown. This problem alone i.e. surface reconstruction from an unordered set of points in space constitutes, on its own, a full area of research. Out of the three criteria chosen, only the distance from the extreme planes could be used without exceptions. The point distribution measure presented some flaws if the points were all clustered in a small region and the flatness gave wrong indications in the case of flat sets not parallel to the reference plane.

Implications for Future Research

Several parts of this work are interesting starting points for further research. First, there is no clear correspondence between the Single Objective CDPs (SOCDPs) complexity and the MOPs complexity. The classification proposed relies on the individual study of the objective functions. However, the size of the Pareto set is not only related to the size of the SOCDPs but also to their relative position in variable space.

Second, although the CDOM is designed to work with any type of displacements, rotations were left aside. Considering rotations in the complexity analysis makes the non-linearity of the problem more sensitive to the shape of each component and requires the introduction of additional factors linked to the non convex aspects of the shapes like holes and shallow parts.

Third, the tessellated surface representations of the solid required more precision. Using an automatic refinement of the tessellation in areas of interest, like the algorithms used for fast visualization in Computer Graphics, might bring more precision in the interference detection. In this same domain, using a quadtree 3D space decomposition will bring a faster interference checking and will speed up the full computation especially for assemblies with many components. Moreover surface representations do not allow the detection of configurations in which a small object is embedded into a large one.

Fourth, using the relative motion variables with respect to a single reference has two drawbacks. First it does not guarantee that the components remain inside the system, and

second it does not allow components to follow complex paths. On the other hand, allowing the engineer to choose the degrees of freedom leaves open the possibility to include knowledge-based systems for incorporating, in the CDP solving process, rules driven by the type of design and the intent of the designer. These rules can then be applied to choose the best design variables in order to achieve the goals. The iterations of the CDOM introduce an additional flexibility in the method that can be used to explore several promising regions of the design variable space. The search can then be easily piloted by higher level rules.

Fifth, complex placement involving several references for the same component was not investigated. It is believed that this case needs a preliminary solving step verifying the compatibility of the position of the object with respect to all its references. A second possibility consists in using again penalty functions, however, in addition to bringing additional complexity to the landscape, this solution is in contradiction with the reasons why the relative placement strategy was proposed.

In the field of Genetic Algorithms the use of population seeding should by tried in order to keep extreme members inside each sets. This should help the solution to spread more widely. Bigger populations must also be used in order to take advantage of the full capabilities of the Genetic Algorithm keeping in mind that each genome is already long (see the Genetic Algorithm Technical Description in Appendix A).

Finally, CDOM provides multiple solutions each corresponding to an optimal configuration. From this set the engineer must choose a single configuration that will be sent to manufacturing. The choice of this final design can rely on subjective criteria like aesthetics. How this choice is made is again an area of research that was not covered by the present work.

Other Implications and Criticisms

In addition to the mechanical engineering and mathematical aspect of this dissertation a large part of the work consisted in programming the CDOM. Part of its implementation is designed according to the Object Oriented programming standards. However, the program which was used for testing the CDOM is a monolithic code which is not designed for easy integration and reuse. Several issues were raised from this bad implementation among which the waste of time spent testing classes after modifications were done and the management of the different versions of the CDOM. Eventually, collecting data for the statistical studies

was a long process subject to errors because of the absence of automatic procedures embedded into the code when it was designed.

Summary of this Dissertation

In the search for an optimization method able to address the placement of freeform components in complex mechanical assemblies, three goals were achieved. First, a method was proposed, bypassing the problem of solving thousands of nonlinear constraints, that finds multiple solutions to the Engineering Configuration Design Problem. Second, a complexity classification is defined applicable to the Configuration problems and based on simple physical characteristics of the assemblies. Third, several quality criteria applicable to the search of unknown Pareto sets were investigated.

APPENDICES

Appendix A

Technical Parameters

Multivariate Functions Trend

To generalize the definition of the Trend to functions with N variables and one global optimum, it is sufficient to extend the definition of the Unit Trend (unitTr) attached to each point of the variable space. It is proposed to modify equation (10) in the following way:

$$\mathbf{unitTr}[\, x\,,\, x_0,\, x_c] = \begin{cases} 1 \text{ if } d[\, x,\, x_c\,] > r \text{ and } \mathbf{F}[x_0] < \mathbf{F}[x]\,; \\ 0 \text{ if } d[\, x,\, x_c\,] > r \text{ and } \mathbf{F}[x_0] > \mathbf{F}[x]\,; \\ -1 \text{ if } d[\, x,\, x_c\,] < r \text{ and } \mathbf{F}[x_0] < \mathbf{F}[x]\,; \\ 0 \text{ if } d[\, x,\, x_c\,] < r \text{ and } \mathbf{F}[x_0] > \mathbf{F}[x]\,; \\ 0 \text{ otherwise;} \end{cases}$$

where x_c is the center of the ball which has the segment $[x_{goal}\,,\, x_0]$ as diameter, and r is the radius of this same ball (Figure A.1), and d[A,B] is the Euclidean distance between point A and point B.

For a sampled variable space, the definition of the point Trend remains unchanged.

$$\mathbf{TrPoint}[\, x_0\,] = \sum_{x \text{ in } \{ BF \}} (\, \mathbf{unitTr}[\, x\,,\, x_0,\, x_c])$$

$$\mathbf{Tr}[\, F\,] = \sum_{x \text{ in } \{ BF \}} (\, \mathbf{TrPoint}[\, x\,]\,) \,/\, (\, \mathbf{Card}[\, \{ B_F \}\,] * (\mathbf{Card}[\, \{ B_F \}\,] - 1) \,/\, 2$$

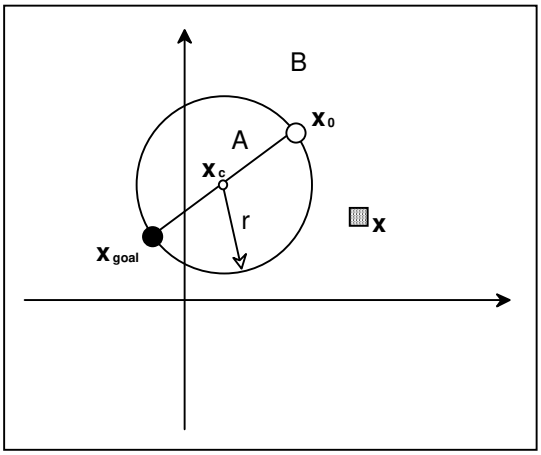

Figure A.1. The unit Trend of x with respect to x_0 depends on **x** is in zone A or B and of the respective value of F[**x**] with respect to F[$\mathbf{x_0}$].

When more than one global optimum is present in the landscape, the unit Trend must be successively calculated for each goal. The Unit Trend of a point x_0 with respect to a current point x is then the sum of the all the Unit Trends resulting from the evaluation of the Unit Trend for each goal.

$$\mathbf{unitTr}[\ \mathbf{x}\ ,\ \mathbf{x_0}\ ] = \sum \mathbf{x_c}\ (\ \mathbf{unitTr}[\ \mathbf{x}\ ,\ \mathbf{x_0}, \mathbf{x_c}\ ]\ )$$

Unimodal x Limit Value for F[x] = a*x - A*Cos[x] + A and Numerical Results

The function F[x] = a*x - A*Cos[x] + A becomes unimodal when its derivative remains null or positive for all x in its definition domain.

For all x in [B_F],

$$\frac{dF[x]}{dx} = a + A * Sin[x]$$

which is positive or null for all x if and only if x r a.

This number matches the results obtained in Chapter III in which the Trend reaches its maximum value for a = 10.

Table A-I. Evolution of the Trend of function F for different slopes (a) A=10, n=1000.

a	0.1	0.25	0.5	0.75	1	2	3	4	5	10	100
Tr	0.19	0.36	0.59	0.70	0.76	0.87	0.90	0.91	0.93	0.99	0.99

Note that the definition of Fp, the p variable version of F, can be changed such that it never becomes unimodal:

$$F_p\,[a,\ \mathbf{x},\ x_0] = \sum_{i=1\ldots p} (a*\mathrm{Abs}[x_i - x_{0i}] - 2a*\mathrm{Cos}[x_i - x_{0i}] + 2a)$$

Description and Numerical Results of GA Test Case 1 (Chapter III p. 100)

The function Abs[x-5] - 10*Cos[x-5] + 10 is given to a Simple GA to find its global minimum over the domain [-2*P*1000, 2*P*1000].

The characteristics of the GA run are :

Population size : 20.

Genome length : 30.

Termination Criterion upon convergence over 20 generations : 0.99999.

Probability of Mutation : 0.005.

Probability of Crossover : 0.6.

Scaling : Sigma Truncation.

Selection : Roulette Wheel.

Table A-II. Evolution of the Average and Standard Deviation of the Number of Generations required for a simple GA to reach convergence calculated over 20 runs for each measure.

Trend	Avg	stdev
0.03	24.85	4.33
0.19	36.05	11.743
0.36	43.7	10.52
0.59	43.75	13.11
0.7	47.25	14.11
0.76	49	15.28
0.87	48.45	14.33
0.9	51.55	18.99
0.91	53.55	16.04
0.93	54.85	11.09
0.99	54.6	14.23
0.99	71.42	25.41

Table A-III. Evolution of the Average and Standard Deviation of the precision achieved after convergence in variable space.

Trend	Avg	Stdev
0.03	0.018	0.011
0.19	0.0023	0.0035
0.36	0.00071	0.00058
0.59	0.00058	0.00049
0.7	0.00049	0.00019
0.76	0.00040	3.6E-05
0.87	0.00040	3.5E-05
0.9	0.00041	8.0E-05
0.91	0.00041	4.4E-05
0.93	0.00041	5.1E-05
0.99	0.00041	4.37E-05
0.99	0.00043	0.00014

Effect of the Unfeasible Area Location on the Trend

For a constant Roughness and Dilution of the feasible area, Equations (8), (9) and (10) show that the Trend is a function of the position of the unfeasible area with respect to the goal (Figures A-2, A-3 and A-4).

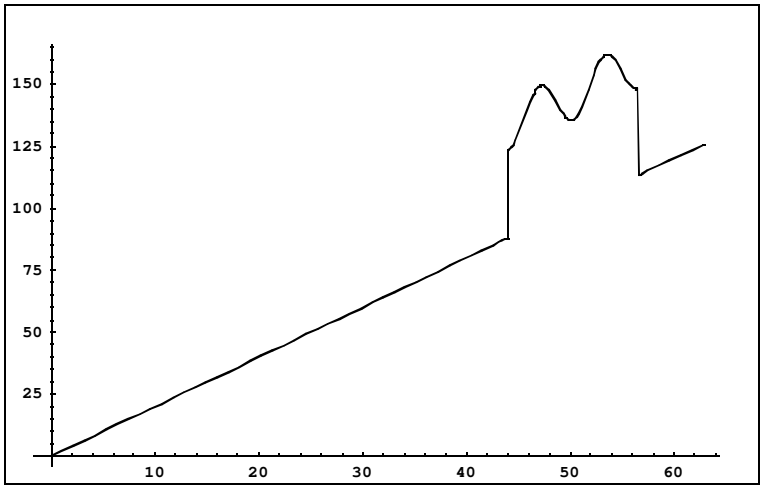

Figure A-2. Function F[x] = 2*x - 10*Cos[x] +35 on]14 Π, 18 Π [and 2*x on [0, 14Π] and [18Π, 20Π]. Trend[F] = 0.91.

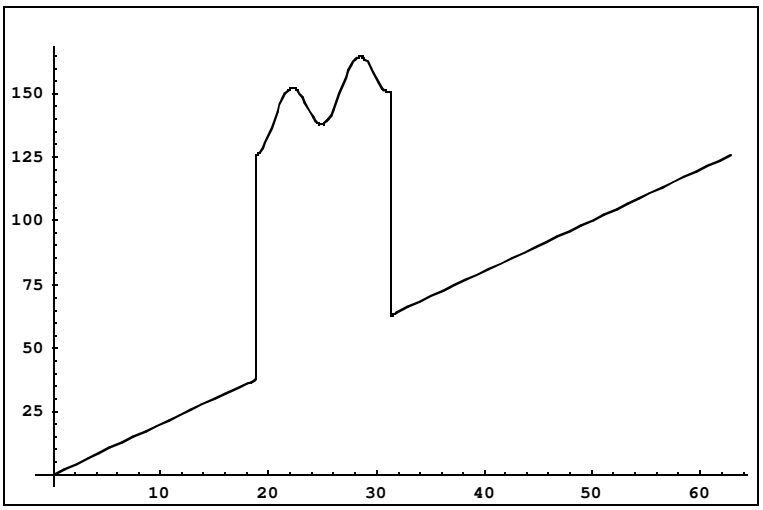

Figure A-3. Function F[x] = 2*x - 10*Cos[x] + 45 on]6Π, 10Π [and 2*x on [0, 6Π] and [10Π, 20Π]. Trend[F] = 0.57.

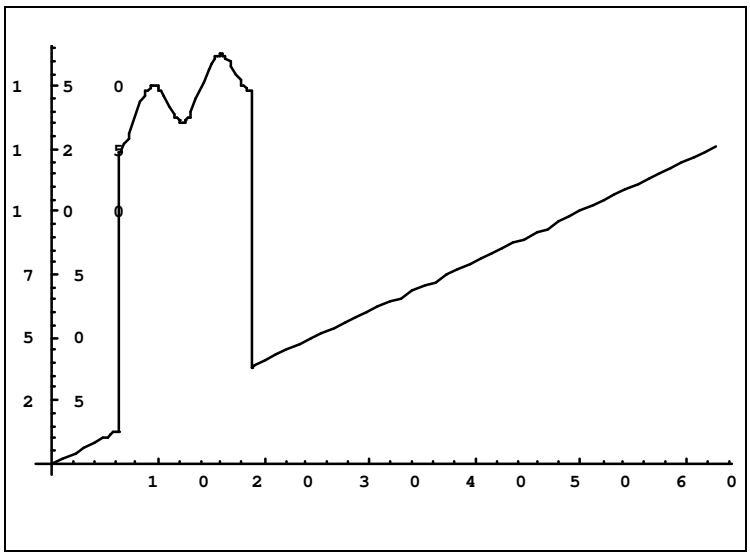

Figure A-4. Function F[x] = 2*x - 10*Cos[x] + 55 on]Π, 5Π[and 2*x on [0, 5Π] and [5Π, 20Π]. Trend[F] = 0.41.

Roughness Expression for Absolutely and Relatively Placed Objects

A CDP involving 'a' objects which can move all over the 3D space (A), and 's' static objects (S), has a roughness equals to:

$$Rg = (1+s)^{a} - (a)*(a-1)/2*[\ ((1+s)^{a-1}-1-s] + (a)*(a-1)*[\ ((1+s)^{a-1}-1-s] - (1+s) + (1+s)*(a)! \qquad (A1)$$

Table A-IV gives the first few values of the Roughness when the number of objects of each category increases.

Table A-IV. Evolution of the Roughness of a CDP involving 'a' objects of category A and 's' static objects.

		a					
		1	**2**	**3**	**4**	**5**	**6**
s	**1**	2	6	24	98	410	1952
	2	3	12	60	294	1380	6486
	3	4	20	120	708	4020	22272
	4	5	30	210	1460	9920	66020
	5	6	42	336	2694	21390	167520
	6	7	56	504	4578	41580	374682

The first term of this expression, $(1+s)^a$, is due to the interference of the A objects with each static objects. Those interference partition the objective/variable space into rectangular valleys called cells (Figure A-5).

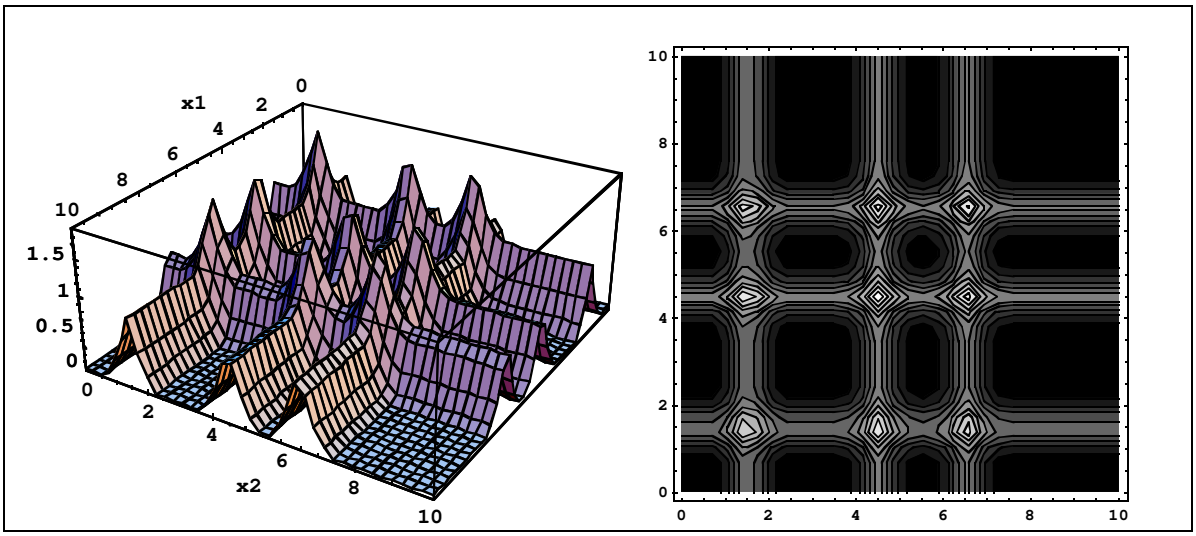

Figure A-5. Landscape of a 5 pallet 1D CDP. Three pallets are fixed and two pallets can move without overlapping with each other, generating a landscape containing 16 valleys.

The second and third terms are due to the interference of each A objects with the other A objects. These interferences have for effect to split all the cells on the diagonal of each hyper-space of degree a-2 into 2 valleys (Figure A-6 compared to Figure A-5).

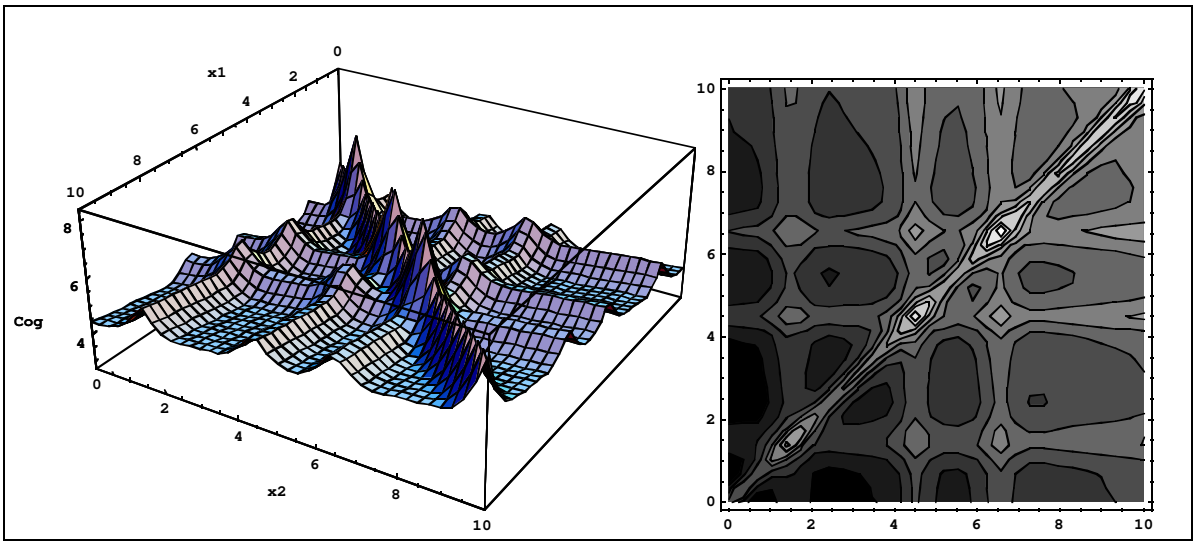

Figure A-6. Landscape of a 2 pallet 1D CDP in which the 2 pallets can move all over the domain and thus can interact, and 3 static pallets.

The fourth and fifth terms are corrections due to the interference of each A objects with the other A objects along the diagonal of the variable space (which is a N variable space).

The 'r' objects which have relative placement with respect to some of the 'a' can interact with all the static objects and with a - 2 other objects. It is assumed that, in the worst case, they cannot interact with their positionner nor with the object they position. Hence, equation (A1) becomes

$$Rg = (1+s)^{a+r-2} - (a+r-2)*(a-1+r-2)/2*[\ ((1+s)^{a-1+r-2}-1-s]$$
$$+ (a+r-2)*(a-1+r-2)*[\ ((1+s)^{a-1+r-2}-1-s] - (1+s) + (1+s)*(a+r-2)! \qquad (A2)$$

which is equal to equation (16).

Genetic Algorithms Technical Descriptions

The genetic algorithm described as SimpleGA is the 'simple' genetic algorithm described in Goldberg (1989). It uses non-overlapping populations. Its main characteristic consists in creating an entirely new population by replacing all the members of the previous one by their children. This process stops when convergence is reached. No Elitism is used.

The scaling method consistently used along all this work is Sigma Truncation.

The selection scheme adopted is the Roulette Wheel Selector which selects an individual with a probability proportional to the magnitude of its fitness score relatively to the rest of the population.

The second GA used in this work is a SteadyStateGA i.e. a GA with overlapping populations. The overlap of each population is 25%.

The population size was of 15 members and the number of designs per child is 15. The genome length is 10. This makes the length of each individual genome equal to 10*(number of variables)*15 bytes. The maximum genome length is thus equal to 2700 bytes (18 variable 3D CDPs).

The stopping criterion is based on convergence. A failure to improve the best fitness of 0.001% stops the GA.

CDOM algorithm

```
Read Input files
Calculate Objects Data
Cog
Inertia Matrices
Mass

While( improvement > e or previous_improvement > e) do
{
        if(improvement>0)
        {
                //c Restricting domain
                restriction_coef = restriction_coef x k1
                penalty_coef = penalty_coef x k2
        }
        else
        {
                //c Expanding domain
                restriction_coef = restriction_coef x k3
        }
        Run GA()
}

RunGA()
{
        variables_bounds = variables_bounds*restriction_coef
        define new genomes
        Evolve( Objective_function() )

}

Objective_Function(cloud)
{
        Calculate objective value for all the members of the current cloud
        Update the reference Pareto set
        Rate the cloud
        Return the Grade
}
```

Appendix B

Input Files and Libraries

Cube CDP

```
4
100.0 100.0 40.0 -20.0 -20.0 -20.0
35.0 35.0 0.0

cube_simple6.stl
cube_simple6.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

10.0
8000.0
0 0
2

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

0.0 0.0 70.1
0.0 0.0 70.1
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

cube_simple6.stl
cube_simple6.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

10.0
8000.0
0 1
0

cube_simple6.stl
cube_simple6.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

10.0
8000.0
0 1
0

cube_simple6.stl
cube_simple6.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

10.0
8000.0
0 1
0

//==============================
//Example of Input
4           <= Number of components
700.0 700.0 150.0 -20.0 -20.0 -20.0 <= bounds of the
system
30.0 30.0 10.0      <= Target center of gravity.


cube_simple.stl <= shape file description

0.0 -10.0 10.0      <= authotrized dof with respect to
the reference position of the reference object.
0.0 -10.0 10.0
0.0 -10.0 10.0
0.0 0.0 6.29
0.0 0.0 6.29
0.0 0.0 6.29

100.0     <= density of the component.
8000.0 <= volume of the component.
1 1                 <= reference object and reference
position

2                   <= number of links attached to this
object

0.0 -999.9 999.9    <= first reference position tx, ty, tz,
rx, ry, rz and type;
0.0 -999.9 999.9
0.0 -999.9 999.9
0.0 -999.9 999.9
0.0 -999.9 999.9
0.0 -999.9 999.9

10.0 -999.9 999.9   <= second reference position tx, ty,
tz, rx, ry, rz and type;
12.0 -999.9 999.9
-5.0 -999.9 999.9
-3.1415 -999.9 999.9
0.0 -999.9 999.9
0.0 -999.9 999.9
```

Satellite CDP

```
2000.0 2000.0 2000.0 -2000.0 -2000.0 -2000.0
0.0 0.0 0.0

sat2.stl
sat2.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

5.0
40.0
0 0
3

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

0.0 -200.0 200.0
0.0 -200.0 200.0
0.0 -150.0 150.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

0.0 -200.0 200.0
0.0 0.0 200.0
0.0 -120.0 -120.1
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

sat5.stl
sat5.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

3.0
2.0
0 1
1

0.0 -200.0 100.0
0.0 0.0 0.0
0.0 -50.0 150.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

sat6.stl
sat6.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

2.57
2.0
1 0
1

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

sat4.stl
sat4.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

2.57
2.0
0 2
1

0.0 0.0 0.0
0.0 -400.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

sat8.stl
sat8.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

2.57
2.0
3 0
1

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

sat1.stl
sat1.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
```

```
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

2.57
2.0
0 1
1

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0


sat7.stl
sat7.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

2.57
2.0
0 0
1

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0


sat9.stl
sat9.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

2.57
2.0
```

```
0 1
1

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0


//==============================
//Example of Input
4            <= Number of components
700.0 700.0 150.0 -20.0 -20.0 -20.0 <= bounds of the
system
30.0 30.0 10.0       <= Target center of gravity.


cube_simple.stl <= shape file description

0.0 -10.0 10.0       <= authotrized dof with respect to
the reference position of the reference object.
0.0 -10.0 10.0
0.0 -10.0 10.0
0.0 0.0 6.29
0.0 0.0 6.29
0.0 0.0 6.29

100.0      <= density of the component.
8000.0 <= volume of the component.
1 1                  <= reference object and reference
position

2                    <= number of links attached to this
object

0.0 -999.9 999.9     <= first reference position tx, ty, tz,
rx, ry, rz and type;
0.0 -999.9 999.9
0.0 -999.9 999.9
0.0 -999.9 999.9
0.0 -999.9 999.9
0.0 -999.9 999.9

10.0 -999.9 999.9    <= second reference position tx, ty,
tz, rx, ry, rz and type;
12.0 -999.9 999.9
-5.0 -999.9 999.9
-3.1415 -999.9 999.9
0.0 -999.9 999.9
0.0 -999.9 999.9
```

Car Engine CDP

```
8
2000.0 2000.0 2000.0 -2000.0 -2000.0 -2000.0
0.0 0.0 0.0

motorBlockOrg.stl
motorBlockOrg.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

3.5
24000.0
0 0
5

0.0 0.0 0.0
0.0 0.0 0.0
```

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

0.0 100.0 400.0
0.0 -400.0 400.0
0.0 -250.0 250.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

0.0 -300.0 300.0
0.0 -190.1 -190.0
0.0 -200.0 200.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

0.0 -50.0 50.0
0.0 -300.1 -300.0
0.0 -100.0 100.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

0.0 -400.0 -100.0
0.0 -400.0 400.0
0.0 -250.0 250.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

batteryOrg.stl
batteryOrg.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

3.0
968
0 1
1

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

waterOrg.stl
waterOrg.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

1.0
968.0
0 4
1

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

fanOrg.stl
fanOrg.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

2.5
924
0 3
1

0.0 -50.0 50.0
0.0 -75.1 -75.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

radiatorOrg.stl
radiatorOrg.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

1.8
3875
3 0
1

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

alternatorOrg.stl
alternatorOrg.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

2.57
2.0
0 2
1

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

```
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0


powerSOrg.stl
powerSOrg.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

2.5
600
0 2
1

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0


access.stl
access.stl.dat

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

2.57
2.0
0 1
1

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
```

```
0.0 0.0 0.0
0.0 0.0 0.0


//==============================
//Example of Input
4            <= Number of components
700.0 700.0 150.0 -20.0 -20.0 -20.0 <= bounds of the
system
30.0 30.0 10.0       <= Target center of gravity.


cube_simple.stl <= shape file description

0.0 -10.0 10.0       <= authotrized dof with respect to
the reference position of the reference object.
0.0 -10.0 10.0
0.0 -10.0 10.0
0.0 0.0 6.29
0.0 0.0 6.29
0.0 0.0 6.29

100.0      <= density of the component.
8000.0 <= volume of the component.
1 1                  <= reference object and reference
position

2                    <= number of links attached to this
object

0.0 -999.9 999.9     <= first reference position tx, ty, tz,
rx, ry, rz and type;
0.0 -999.9 999.9
0.0 -999.9 999.9
0.0 -999.9 999.9
0.0 -999.9 999.9
0.0 -999.9 999.9

10.0 -999.9 999.9    <= second reference position tx, ty,
tz, rx, ry, rz and type;
12.0 -999.9 999.9
-5.0 -999.9 999.9
-3.1415 -999.9 999.9
0.0 -999.9 999.9
0.0 -999.9 999.9
```

Example of Additional Data File

The shape of each object is defined by an stl file already defined in Chapter III. Other mechanical characteristics such as the inertia matrix are put in separate file identified by its extension ".dat".

```
180 <= diameter of the danger zone
0.0 0.0 0.0 <= center of the object (usually the origin from which are calculated the coordinates of the stl file triangles)

// Inertia matrix.
1000 0 0
0 1000 0
0 0 1500
```

External Libraries Used by the CDOM

GaLib 2.4.1

A Genetic Algorithm library written in C++ at the MIT CADLAB by Matthew Wall. This library can be accessed using the Encore web site at http://alife.santafe.edu:80/~joke/encore/

RAPID

A fast intersection checking library for tessellated solids developed in C at the University of North Carolina at Chapel Hill Computer Science Department by S. Gottshalk.

CUGL3D

A library written in C++ at Clemson University by Donal G. Owens and Pierre Grignon, offering basic 3D geometric objects classes such as points, lines, polygons, spheres as well as 3D transformations (translations, rotations, scaling).

BIBLIOGRAPHY

(1995). Encyclopedia Britannica.

ALS (1997). General Geometric Constraint Manager V 1.0, ALS.

Amiouny, S. V., Bartholdi, J. J. III, Vande-Vate, J. H. and Zhang, J. (1992). "Balanced Loading." Journal of Operational Research: 238-246.

Aziz, N. M. (1991). "A Computer-aided Box Stacking Model for Truck Transport and Pallets." Computers in Indutry.

Baker (1985). Adaptive Selection Methods for Genetic Algorithms. First International Conference on Genetic Algorithms and Their Applications, Pittsburg, Lawrence Erlbaum Associates Hillsdale, New Jersey.

Baldwin, N. (1995). "The Laboratory Notebooks of Thomas Edison." Scientific American(October).

Beasley, D., Bull, D. R., Martin, R. R. (1993). "A Sequentiel Niche Technique for Multimodal Function Optimization." Evolutionary Computation 1(2): 101-125.

Becker, R. W., and Lago, G. V., (1970). Proceedings of the 8th Allerton Conference on Circuits and Systems Theory, Allerton.

Benson, H. P. (1978). "Existence of efficient solutions for vector maximization problems." Journal of Optimization and Applications(26): 569-580.

Bertanlaffy (1874).

Brent, R. P. (1973). Algorithms for Minimization without Derivatives. Englewood Cliffs, NJ,, Prentice Hall,.

Campbell, M. J., Amon, C. H., Cagan, J. and Sykman, S. (1995). "Electronic component Placement using SA under Thermal Constraints." ASME IMECE.

Carlson, S. E., and Pegg, T. A. (1995). "Component Selection using Genetic Algorithms." Advances in Design Automation 1: 65.

Chankong, V., and Haimes, Y. Y. "Multiobjective Decision Making: Theory and Methodology." 8.

Chapman, C. D., Saitou, K., and Jakiela, M. J. (1994). " Genetic Algorithms as an Approach to Configuration and Topology Design." Journal of Mech. Design 116(1005).

Chevalier, A. (1986). Guide du dessinateur industriel, Hachette.

Coffignal, G. (1992). "Mecanique." (Cours de 3eme annee a l'ecole Nationale Superieure des Arts et Metiers, Paris).

Corcoran, A. L., and Wainwright, R. L. (1992). A genetic Algorithm for Packing in Three Dimensions. SAC'92 Proceedings of the 1992 ACM/SIGAPP Symposium, Kansas City, ACM Press NY.

Council, N. R. (1991). Improving Engineering Design. Designing for Competitive Advantage, National Academy Press.

Croft, T. H., Falconner, J. K., and Guy, R.K. (1991). Unsolved Problems in Geometry, Springer-Verlag.

D-Cubed. (1998). Dimensinal Constraint Manager, D-Cubed Ltd. 68 Castle Street, Cambridge, CB 3 0AJ, England.

Dassault Systemes, 9 Quai Marcel Dassault, Suresnes, France.

DeJong, K. A. (1975). Analysis of a Class of genetic Adaptative Systems, University of Michigan.

Descartes, R. (1628). Le Discours de la Methode.

Dixon, J. R., Mowe, A., Cohen, P. R., and Simmons M. K. (1986). "DominicI: Progress Towards Domain Independence in Design By Iterative Redesign." Computers in Engineering: 199-206.

Dixon, J. R., and Finger, S. (1989). "A Review of Research in Mechanical Engineering Design. Part I: Descriptive, Prescriptive, and Computer-Based Models of Design Process." Research in Engineering Design: 51-67.

Dixon, J. R., and Finger, S. (1989). "A Review of Research in Mechanical Engineering Design. Part II: Representations, Analysis, and Design for the Life Cycle." Research in Engineering Design: 121-137.

Dixon, J. R., and Duffey, M.R. (Winter, 1990). The Neglect of Engineering Design. Mechanical Design Automation Technical Report 5.89, Berkeley, CA.

Dowsland, K. A., and Dowsland, W. B. (1992). "Packing Problems." European Journal of Operational Research(56): 2-14.

Dowsland, K. A. (1993). “Some Experiments with Simulated Annealing Techniques for Packing Problems.” European Journal of Operational Research(68): 389-399.

Dyckoff, H. (1990). “A Typolgie of Cutting and Packing Problems.” European Journal of Operational Research(44): 145-159.

Ferguson, E. S. (1977). “The Mind's Eye: Nonverbal Thought in Technology.” Science 197(4306).

Fonseca, C. M., and Fleming, J., (1993). Genetic Algorithms for Multi Objective Optimization, Formulation, Discussion and Generalization. Proceedings of the Fifth Internationql Conference on Genetic Algorithms, Urbana Champaign, IL, Morgan Kaufman, San Mateo, CA.

Gardner, M. (1979). “Mathematical Games: Some Packing Problems Cannot be Solved by Sitting on the Suitcase.” Scientific American 241(4).

Garey, M. R., and Jonhson, D. S., (1979). Computers and Intractability. San Francisco, Freeman.

Gerald, F. (1990). Curves and Surfaces.

Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. Reading, MA, Addison-Wesley.

Golomb, S. W. (1970). Polyominoes. Princeton.

Graham, R. L. (1978). “The Combinatorial Mathematics of Scheduling.” Scientific American 239(3): 124.

Grignon, P. M., Wodziak, J. R, and Fadel, G. M. (1996). “Bi-Objective Optimization of Components Packing Using a Genetic Algorithm.” Multidisciplinary Analysis and Optimization, AIAA.

Grignon, P. M., and Fadel, G. (1997). Quality criteria measures for multi objective solutions Obtained with a Genetic Algorithm. AIAA SDM Conference, Orlando.

Haftka, R. T., and Gurdal, Z. (1993). Element of Structural Optimization, Kluwer Academic Publishers.

HarVest (1997). Le Journal Des Bureaux d'etudes: 16.

Hogg, T., and Huberman B. A. (1992). Better Than the Best: The Power of Cooperation. 1992 Lectures In Complex Systems. L. Nadel, and Stein, D. L. Santa Fe, Addison-Wesley. V: 3-88.

Holland, J. H. (1975). Adaptation in Natural and Artificial Systems. Ann Harbor, The University of Michigan Press.

Hsuan, C. (1996). "A New Technnique to Handle Local Minimum for Imperfect Potential Field base Motion Planning." IEEE ICRA.

Huang Min, W. a. A., J. S. (1995). Engineering Optimization with Discrete Variables. 36th AIAA/ASME/ASCE/AHS/ASC Structures Dynamics and Material Conference and AIAA/ASME Adaptative Structures Forum, New Orleans, LA.

Jain, S. a. G., H. C. (1995). "PCB Layout Design Using a GA." Advances in Design Automation 1: 529-536.

Julesius, C. (1989). A Topographical Method for Global Optimization. Sweden, Abo Akademi.

Kamal, S. Z., Karandikar, H. M., Mistree, F. and Muster, D. (1987). "Knowledge Representation for Discipline- Independent Decision Making." Expert Systems in Computer-Aided Design: 289-321.

Kane, C., Jouve, F. and SChoenauer, M. (1995). "Structural Topology Optimization in Linear and Non Linear Elasticity Using Genetic Algorithms." Proceedings of the 1995 Design Engineering Technical Conferences 1, DE-Vol, 82: 385=392.

Karnopp, D., Margolis, D., and Rosenberg, R. (1990). System Dynamics-A Unified Approach. N.Y.

Kim, J. J., and Gossard, D. C. (1991). "Reasoning on the Location of Components for Assembly Packaging." Journal of Mechanical Design 4(113): 402-407.

Kirkpatrick, S., Gelatt, C. D. and Vecchi, M.P. (1983). "Optimization by Simulated Annealing." Science 220(4598): 671-680.

Knecht , H. (1994). "Late Ice Age Hunting Technology." Scientific American 66: 82.

Ko, K. (1991). Complexity Theory of Real Functions. Boston.

Kostreva, M. M., and Wiecek, M. M. (1993). "Linear Complementarity Problems and Multiple Objective Programming." Mathematical Programming, 60, 349-359.

Kreisselmer, G., and Steinhauser R. (1979). Systematic Control Design by Optimizing a Vector Performance Index. Proceedings of IFAC on Computer Aided Design of Control Systems, Zurich, Switzerland.

Lewis, H. R., and Papadimitriou, C. H. (1978). "The Efficiency of Algorithms." Scientific American(January): 96-109.

Lomangino, P. F. (1994). Grammar and Optimization-Based Mechanical Packaging, Georgia Institute of Technology.

Macken, C. A., and Stadler, P. F., (1993). Evolution of Fitness Landscape. Lectures in Complex Systems. L. Nadel, and Stein, D. L.,. Santa Fe, Addison-Wesley. VI: 43-85.

Mahfoud, S. W. (1995). Niching Methods for Genetic Algorithms. Urbana-Champaign, of Illinois: 264.

Mistree, F., Smith, W. and Bras, B. (1993). A Desicion Based Approach to Concurrent Design. Handbook of Design and Manufacturing, Chapman and Hall.

Mitchell, M. (1992). Genetic Algorithm. 1992 Lectures In Complex Systems. L. Nadel, and Stein, D. L. Santa Fe, Addison-Wesley. V: 3-88.

Moscato, P. (1989). On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Toward Memetic Algorithms. Pasadena, CA, CalTech.

Nakayama, W. (1990). "On the Accomodation of Coolant Flow Paths in High Density Packaging." IEEE Transactions On Components, Hybrids, and Manufacturing Technology 13(4).

O'Rourke, J. (1993). Computational Geometry in C. Cambridge, Cambridge University Press.

Oei, C. K., Goldberg, D.E., and Chang, S. J. (1991). Tournament Selection, Niching, and the Preservation of Diversity. Urbana Champaign, Illinois, University of Illinois at Urbana Champaign.

Orelup, M. R., Dixon, J. R., and Simmons M. K. (1987). Dominic II: Progress Towards Domain Independence in Design By Iterative Redesign. ASME Winter Annual Meeting, Boston, MA,.

Owen, G. (1996). The Dimensional Constraint Manager Version 7.0. Cambridge, D-Cubed LTD, 68 Castle Street, CB3 0AJ.

Packel, W. E., and Traub, J. F., (1987). "Information-based complexity." Nature 328(July): 29-33.

Papadimitriou, C. H., and Lewis, H. R., (1978). "The Efficiency of Algorithm." Scientific American(January): 96.

Powel, M. J. D. (1985). "An Efficient Method for Finding the Minimum of a Function of Several Variables Without Calculating Derivates." Computer J.(7): 155-162.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W.T.. (1990). Numerical recipes in C, the art of scientific computing, Cambridge University Press.

Prigogine, I., and Nicolis, G. (1989). Exploring Complexity. New York, W. H. Freeman and Company.

Ramachandran, N., Shah, A. H., and Langrana N. A. (July 1988). "Expert System Approach in Design of Mechanical Components." Journal of Engineering with Computers(4): 185-195.

Richardson, J. T., Palmer, M. R., Liepins G., and Hillliard, M. (1989). Some Guidelines for Genetic Algorithm with Penalty Functions. Proceeding of the Third International Conference on Genetic Algorithms, George Masson University, Morgan Kaufmann, San Mateo, CA.

Rosen, J. B. (1960). "The Gradient Projection Method for Non Linear Programming Part I : Linear Constraints." The Society for Industrial and Applied Mechanics Journal 8(1): 181-217.

Rosen, J. B. (1960). "The Gradient Projection Method for Non Linear Programming Part II : Non Linear Constraints." The Society for Industrial and Applied Mechanics Journal 9(4): 514-532.

Rowel, A. A. (Sept. 1997). Prototyping in a Digital World. Computer Graphics World: 55-62.

Schaffer, J. D. (1984). Some Experiments in Machine Learning Using Vector Evaluated Genetic Algorithms.

Schultz, A. S., Shmoys, D. B., and Williamson, D. P. (1997). "Approximation Algorithms." Proc. Nat. Acad. Sci. USA, 94, 12734-12735.

Sedgewik, R. (1983). Algorithms, Addison Wesley.

Shaffer, J. D., and Grefenstette, J. J. (1985). "Multiobjective Learning via GA." Proc. 9th Int. J. Conf. Artif. Intel.: 593-595.

Shearer, J., and Kulakowski, B. (1990). Dynamic Modeling and Control of Engineering Systems. N.Y., Mac Millan.

Shrijer, A. (1982). Packing and Covering in Combinatorics, Mathematical Centre Act.

Shroeder, W. J., Langan, J. A., and Linthicum, S. "Designing for Maintainability." .

Sobol, I. M. (1967). "On the distribution of points in a cube and the approximate evaluation of integrals." USSR Comput. Math. math. Phys. 7: 86-112.

Sobol, I. M. (1990). A Global search for multicriterial problems. 9th International conference on Multiple Criteria Decision Making, Washigton.

Statkinov, and Matusov. (1995). Multicriteria Optimization and Engineering., Capman & Hall.

Stewart, I. (1993). "Packing Problems in a Sport-gear Shipping Room." Scientific American: 142.

Sthanusubramonian, T., Finger, S. and Rinderle, J. (1992). A Transformatorial Approach to Configuration Design. Proccedings of the 1992 NSF Design and Manufacturing Systems Conference, Georgia Institute of Technology, Atlanta, GA.

Stoyan, Y. G. (1983). "Mathematical Methods for Geometry Design." Advances in CAD/CAM.

Szykman, S., and Cagan, J. (1994). A simulated Annealing-Based Approach to 3D Component Packing. Advances in Design Automation 1994: Proceedings of the 20th ASME Design Automation, Mineapolis,MN.

Szykman, S., and Cagan, J. (1994). "Contained 3D Components Layout Using Simulated Annealing." Concurent Product Design ASME DE-Vol. 74.

TenHuisen, M. L., and Wiecek, M. M. (1996). "An Augmented Lagrangian Scalarization for Multiple Objective Programming." Advances in Multiple Objective and Goal Programming, Springer, 151-159.

Torn, A., and Viitanen, S., (1992). Topographical Global Optimization. Recent Advances in Global Optimization. C. A. F. a. P. M. Pardalos. Princeton, Princeton University Press. 1: 384-398.

Traub, J. F. (1994). "Breaking Intractability." Scientific American: 120.

Ullman, D. G. (1992). The Mechanical Design Process, McGraw-Hill.

Ullman, D. G. (1993). "The Evolution of Function and Behavior During Mechanical Theory and Methodology." Design Theory and Methodology, ASME: 91-103.

Venugopal, V., and narendran, T. T. (1992). "A Genetic Algorithm Approach to the Machine-Component Grouping Problem with Multiple Objectives." Computers and Industrial Engineering 22(4): 469-480.

Viitanen, S. (1991). SPATRA, a Topographical Method for Parallel Global Optimization. Sweden, Abo Akademi.

Weiner, N. (1948). Cybernetics. Paris.

Wodziak, J. R. (1994). Optimal Packing Utilizing Genetic Algorithms. Mechanical Engineering. Clemson, Clemson University.

Womack, J. P., Jones, D.T., and Roos, D. (1991). The Machine That Change the World, HarperPerennial.

Zadeh, L. A. “Optimality and non-scalar-valued performance criteria.” IEEE Transactions on Automatic Control(AC-8): 59- 60.