

Knowledge-based Methods for Integrating Carbon Footprint Prediction Techniques into New Product Designs and Engineering Changes

by

Seung-Cheol Yang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Mechanical Engineering)
in The University of Michigan
2010

Doctoral Committee:

Adjunct Professor Debasish Dutta, Co-chair

Lalit M. Patil, Co-chair

Associate Professor Steven J. Skerlos

Assistant Professor Clayton D. Scott

Lakshmi Y. Srinivas, Ford Motor Company

© Seung-Cheol Yang
All Rights Reserved
2010

To my family, friends and teachers.

ACKNOWLEDGEMENTS

First, I would like to thank my co-advisor, Prof. Debasish Dutta, for his support and guidance of my dissertation work. His constructive comments and insightful advice were always encouraging and motivating to me.

I am grateful to my co-advisor, Dr. Lalit Patil. His consistent aids impacted my successful completion of doctoral research. Whenever I encountered research concerns and troubles, I felt assurance talking to Lalit. He was a great source of new ideas presented in my dissertation work.

I would also like to appreciate my committee members: Prof. Skerlos, Prof. Scott, and Dr. Srinivas, for their insightful comments and ideas on my research.

I would like to acknowledge the financial support from National Science Foundation (NSF) and PLM Alliance.

I appreciate all the help and the support I received from group members Yeo IL and Chandresh Mehtar.

Finally, I am grateful to my family and all my friends who physically and mentally support me to get through difficult times. This work would not have been possible without the support and encouragement of my parents, my sister, and friends.

TABLE OF CONTENTS

DEDICATION	ii
ACKNOWLEDGEMENTS	iii
LIST OF TABLES	vii
LIST OF FIGURES	viii
CHAPTER	
I. Introduction	1
1.1 Background	1
1.2 Methodologies for product carbon footprint calculation	2
1.3 Carbon footprint estimation in early design decision	4
1.4 Knowledge based approach to carbon footprint prediction	7
1.4.1 Need to define similar cases in new product design and engineering change	8
1.4.2 Carbon footprint estimation with incomplete infor- mation	10
1.5 Research objective	11
1.6 Outline of the dissertation	12
II. Literature review	14
2.1 Representation of function	14
2.2 Similarity measure for engineering changes	16
2.3 Carbon estimation in the early phase of design	17
2.4 Summary	18
III. Function representation for understanding product semantics	19
3.1 Motivation	19
3.2 Objective	20
3.3 Modeling the Function Semantics Representation (FSR)	20
3.3.1 The core OWL ontology	21

3.3.2	Semantic Web Rule Language (SWRL) rules in the FSR	26
3.4	Case Studies	31
3.4.1	Consistency Checking	32
3.4.2	Semantic retrieval	34
3.5	Summary	37
IV.	Clustering Engineering Changes	39
4.1	Motivation	39
4.2	Objective	41
4.3	Engineering change data structure	41
4.4	Similarity measure for engineering changes	45
4.4.1	EC similarity measure using alignment matching	45
4.5	Parallel engineering change clustering	53
4.5.1	Clustering method for engineering changes	53
4.5.2	Parallel engineering change clustering algorithm	54
4.6	Evaluation	59
4.6.1	Evaluation of similarity computation	59
4.6.2	Evaluation of overall clustering approach	64
4.7	Summary	66
V.	Predicting carbon footprint of proposed new products and engineering changes	68
5.1	Motivation	68
5.2	Environmental strategy for developmental process	69
5.3	Carbon footprint estimation for environmentally friendly products	70
5.4	Fast prediction of carbon footprint for a new product design and a proposed engineering change	70
5.5	Knowledge-based carbon footprint prediction	72
5.6	Overall approach	73
5.7	Detailed procedure of knowledge-based carbon footprint classification	75
5.7.1	Information from concept design and engineering changes	75
5.7.2	Key activity parameters	77
5.7.3	Estimator for key activity data of proposed engineering changes and concept design	79
5.7.4	Estimating key parameters of concept design and proposed engineering changes	82
5.7.5	Carbon footprint classification using previously evaluated carbon footprint and key parameters	86
5.8	Evaluation	89

5.8.1	Case Study	91
5.9	Summary	96
VI.	Conclusion	98
6.1	Research contributions	98
6.2	Future research directions	100
Bibliography	103

LIST OF TABLES

Table

3.1	Number of input and output flows may not always be used to differentiate functions from one another.	25
4.1	Product features over product lifecycle	44
4.2	Precision and recall (in percentage) for the 91 matching problems in the database shown in the appendix.	63
4.3	T-test between the difference of carbon footprint between ECs and pairs of ECs that have an EC similarity value over 0.90	64
4.4	The results of comparative study using F-measure for a cluster	66
5.1	Available information in the design process adopted from [1, 2]	76
5.2	The prediction of carbon significance on new products and proposed ECs	96

LIST OF FIGURES

Figure

3.1	Representing disjoint Flows in RDF/XML.	22
3.2	Every individual flow is different from every other flow.	22
3.3	Partial RDF/XML code for definition of Function.	23
3.4	Partial taxonomy of the FSR.	24
3.5	Definition of hasMagnitude in OWL. Some terms are in bold to highlight important elements in the definition	26
3.6	Counting the number of properties using <code>abox:hasNumberOfPropertyValues</code> for the function <i>Import</i>	28
3.7	Partial RDF/XML syntax showing SWRL rule for the function <i>Import</i>	29
3.8	Using <code>abox:hasClass</code> to determine type of <i>Flow</i> in identification of the function <i>Signal</i>	29
3.9	Using <code>tbox:sameAs</code> to detect the same types of flows in the function <i>Distribute</i>	30
3.10	Using <code>tbox:differentFrom</code> to detect the different types of flows in the function <i>Convert</i>	31
3.11	Using SWRL built-ins to compare values of different properties and detect the function <i>Position</i>	31
3.12	Partial functional model of a <i>Relay</i> used as a part of an ABS Brake System.	33
3.13	Result of consistency checking indicates that the function, <i>Actuate</i> of the <i>Relay</i> in Figure 3.12 is invalid.	35

3.14	Partial functional model of a hypothetical water kettle used to demonstrate semantic retrieval of functionally similar products	35
3.15	Semantic Query-Enhanced Web Rule Language (SQWRL) query to search a product containing <i>Convert</i> , <i>Change</i> , and <i>Store</i> function (so that it is similar to our hypothetical water kettle)	36
3.16	Functionally similar instances retrieved into the SQWRL Query Tab in Protégé	36
4.1	Partial schematic of an Engineering Change.	43
4.2	Example of engineering changes used to explain the determination of similarity	46
4.3	Concept used in the calculating similarity ECs	47
4.4	Calculating attribute match value as a part of activation process . .	50
4.5	Schematic of parallelization in EC clustering	58
4.6	A proposed EC and top 2 similar ECs retrieved from database . . .	60
4.7	Results from similarity obtained for query EC (Figure 4.7 (a)) . . .	62
4.8	Correlation of EC similarity with carbon footprint for the 91 pairs of matching problems	62
4.9	Efficiency of core-usage and speed of parallelized EC clustering. Better speeds are obtained with more cores, but the core-usage efficiency decreases; one reason is that the dataset that we used is not large enough	66
5.1	The process of case-based reasoning	73
5.2	Flow chart of knowledge-based carbon footprint classification	74
5.3	Functional model for brake system	81
5.4	Material matrix	82
5.5	Product and engineering changes that have the same material composition	84

5.6	Manufacturing energy adaptation for a brake rotor	85
5.7	0.632 Bootstrap error rate with $B = 200$ for EC carbon data	91
5.8	Functional model of brake caliper assembly and semantic retrieval using FSR	93
5.9	Retrieved examples of caliper from semantic retrieval and material matrix	93
5.10	Adapting process of manufacturing energy consumption	95

CHAPTER I

Introduction

1.1 Background

Sustainable product development has challenged companies that pursue traditional cost-oriented development. Growing interests in ecology and environmental regulations force companies to take environmental impact into consideration. However, traditional designers are inexperienced in environmental impact evaluation and few specific tools exist that provide information related to sustainable product development.

For sustainable development, designers and stakeholders need to factor in the environmental impact with the traditional product development process over the product life cycle; this encompasses numerous engineering decisions about product concepts, process planning and engineering changes. Among the environmental impacts, climate change by Green House Gas (GHG) has been a key issue for industry, government, and international organizations. Product carbon footprint is the metric of GHG, including direct emission from fossil fuel combustion and indirect emission from energy consumption and material flow over the life cycle.

Legislative regulations and international agreements have been prepared to enforce the effort to reduce carbon footprint. These activities provide companies with an opportunity to lead a new market with a cap and trade system or carbon tax. The volume of the carbon market, estimated at US\$126 billion in 2008 [3] continues to grow. Therefore, a method to reduce carbon footprint is essential in product design decision making.

In order to support carbon reduction, product carbon footprint data needs to be identified in terms of problematic parts, material, or processes. The high contribution parts should be improved by redesigning parts or processes. To achieve low product carbon footprint, stakeholders must be informed about the significance of carbon footprint on design decisions.

1.2 Methodologies for product carbon footprint calculation

Products spend natural resources and produce emissions over their life cycle. The need for sustainable development leads to regulations of environmental impact. In order to reduce product emissions, it is critical to identify carbon footprint over the product lifecycle. Currently, several methods have been proposed to assess carbon emission over the product lifecycle.

Life-Cycle Assessment (LCA) is an analytic method to assess environmental impact of products over the life cycle. LCA is based on quantitative analysis of total impact of raw material, manufacturing, use phase and disposal. LCA considers a number of impacts beyond carbon footprint including ozone layer depletion, acid-

ification, human toxicity, and photochemical smog. The 14040 series explains the detailed method of LCA, which includes four steps: goal and scope definition, lifecycle inventory analysis, lifecycle impact analysis, and interpretation. LCA takes a holistic approach to the environmental impact of products or processes and provides reliable carbon footprint assessment. The result of LCA is useful in providing a good understanding of the cause and effects of environmental impact. However, there is no single method to define system boundaries for product carbon footprint assessment. Subjective boundary definition may result in different studies for the same product.

Publicly Available Specification (PAS) 2050 was published by the British Standard Institute (BSI) in 2008; it is an international standard to assess the lifecycle greenhouse gas of products. PAS 2050, based on lifecycle assessment, is a simplified LCA approach. However, PAS 2050 specifies the system boundary with respect to product carbon footprint, and provides clear requirements for system boundary setting. PAS clarified that the input flow rate is more than 1% of the lifecycle GHG gas and total excluded gas is less than 5% of total lifecycle GHG. This approach can provide an understanding of carbon footprint over the product lifecycle. Nevertheless, because PAS 2050 cannot recognize specific issues related to individual products, Product Category Rules (PCRs), which are reference rules for products in a specific categories, is attached to PAS 2050.

Different simplified methods have been proposed with streamlined LCA. Full LCA analysis requires efforts to gather all related inventory information like materials, processes and energies. The cost and complexity of LCA has been a barrier to

applying it to design processes. For this reason, a streamlined LCA method has been an alternative to reduce time and cost. Some strategies for streamlined LCA are as follows: removal of life cycle phase, specific inventory parameters, use of qualitative method, use of surrogate LCI, and screening constituents. Recently, the development of software systems to support carbon footprint estimation enables integrating design tools with carbon footprint analysis. Some researchers have provided a tool to estimate carbon footprint using a computer aided design (CAD) tool and evaluating the carbon weight of manufacturing processes. This simplified method facilitates easy analysis of carbon footprint to allow design modification for reducing carbon impact.

Once detailed design is completed, carbon footprint calculating tools can be used to analyze the GHG emission of products over their lifecycle. Currently there exist a few tools that facilitate carbon footprint calculation in the embodiment design phase. Thus, the recent research trends of carbon reduction efforts extend the applicable domain to the early design phase in order to consider the carbon emission of various design concepts or embodiments.

1.3 Carbon footprint estimation in early design decision

Carbon footprint calculation has been a potential design tool for the environmental needs of GHG emission reduction over the lifecycle. Until now, carbon footprint calculating methods have been used to estimate total carbon emission of existing products and processes. Estimated carbon footprint has been used to identify prod-

ucts or processes generating high GHG emission and to compare the different GHG emissions of similar products. As shown in Section 1.2, tools for carbon footprint calculation can be sufficient in identifying carbon emission in cases where inventory data and calculating resources are available for an existing product. However, due to the demands for a supportive tool to develop low GHG emission products, there is a need for a specific tool able to identify product carbon footprint for a virtual product with incomplete information. Because 80% or 90% of product design and process design are determined during the design phase, the effort to reduce GHG emission can highly influence environmental design when it is performed in the early design process. There are two applications of carbon footprint estimation to environmental design process: new product design and engineering changes.

New product design: New product design is a process of requirement realization with time and cost constraints. The typical development process consists of four phases: needs identification, concept design, embodiment design, and detailed design. Needs identification phase defines functional requirements and design constraints that characterize the scope of product specification. The other three phases realize design parameters (e.g. dimension, form, material, process, etc.) at different levels from virtual concept to specific products. During this process, designers are expected to consider environmental needs in early design decisions. Recent tools integrated with computer aided design (CAD) systems yield carbon footprint estimates when embodiments and manufacturing information are provided. However, at

the concept phase products, forms, and processes are not exactly defined. Current tools of carbon footprint estimation cannot analyze design concepts with incomplete information. Design information in the concept design phase varies with functional requirements, material composition, and weight estimation. Such design information cannot be applied to estimate the carbon footprint of products using current tools.

Engineering changes: Engineering Change (EC) refers to any changes to shape, dimensions, material, etc. of a part or assembly after the initial design has been released that are considered inevitable for product development. An EC enables the change of existing products to improve their environmental quality. Ideally, the carbon footprint of ECs should be estimated with the difference between initial products and changed products. The changed product information is not completed until the engineering change process is finished. Because the engineering change is typically a time-consuming and costly process due to the lifecycle-wide impact estimation, it cannot be repeated to find an environmentally friendly solution of ECs. Recently, methods to estimate the impact of change have been proposed using statistical methods or dependency modeling. However, these impact estimation methods cannot be used with current tools for estimating the carbon footprint of ECs. To reflect the need of carbon reduction in the process of ECs, the carbon emission of any proposed EC, which is an initial request for change, should be estimated.

1.4 Knowledge based approach to carbon footprint prediction

Integrating carbon reduction efforts with product development can be achieved when engineering decisions incorporate carbon footprint considerations. In particular, company strategy involves decisions in the early design phase and decisions during engineering changes. Company designers can achieve reduced carbon footprint for a new product design by considering alternative decisions on material selection, process substitution, and design layout. Product redesign can contribute to carbon reduction with engineering change of environmentally problematic parts.

To estimate carbon emission in design decisions, well-informed product information over the product life cycle is required. However, new product designs and engineering changes only provide incomplete information that is not well refined about virtual products and virtual changes, as shown in Section 1.3. In addition, carbon estimation needs be performed with minimal delay and cost. However, the traditional tools for environmental impact assessment are inappropriate in performing this analysis in the design phase and during engineering change evaluation.

This requirement can be resolved by using previous carbon footprint data. The overall framework is based on Case-Based Reasoning (CBR) assumes the similarly designed products have similar carbon footprints. Case-based reasoning is a problem solving technique to reuse previous knowledge. The method requires the existence of established knowledge system allowing retrieval of similar product information about resource consumption over the product life cycle. In addition, an adaptation method is required to predict the significance of carbon footprint in terms of similar products

and similar ECs.

This knowledge-based prediction for new products and proposed ECs has three issues. The first issue is the uncertainty of product realization on design parameters. Another issue is the need to develop the method to enable fast prediction of carbon footprint. The third issue is compatibility in applying the method to a concept design and to proposed engineering changes. For reuse of previous knowledge, we retrieve knowledge of similar cases from a database and adjust concept designs and proposed ECs into the knowledge-based prediction.

1.4.1 Need to define similar cases in new product design and engineering change

To approximate carbon footprint in early design decisions, one possibility is to use approximate measure using previous carbon footprint knowledge. In practice, knowledge within similar products is usually adequate for providing feasible carbon emission of design options. Hence, retrieving similar cases is the critical issue. Similar cases with domain dependent values are identified through data representation and semantics.

Functional representation for concept design

A design process has been achieved by developing product concepts that divide and aggregate product function into specific components. Traditionally, geometry or shape has been the focus of product data representation and exchange. New tools

that support different phases throughout the product life cycle are being proposed and developed. These consider not only the "what," but also the "how" and the "why" of a design. Product information consists of several different components that are created, used and shared in different phases of the product lifecycle. The primary goal of product development is to create an artifact satisfying a certain function. Multiple researchers have focused on defining the function of a product. This paper uses the interpretation that function is a product's intended purpose. Defining or understanding a product in terms of its function is important in product development and facilitates a wide variety of tasks that include design synthesis, modeling, and lifecycle analysis to evaluate its sustainability. Thus, there is a need to capture and describe product functions in such a way that the products's meaning is explicitly interpretable by software processes so that the different phases and tools used within the product lifecycle can consider it for effective decision-making.

Product function is not only required to represent the product design, from concept design to detailed design, but it can also provide a basis for the retrieval system for product knowledge. Therefore, it should be implemented on the basis of system architecture like PLM (product life-cycle management) that enables stakeholders to access required knowledge over product life cycle.

Similarity measure for EC

Evaluation of engineering changes involves assessing the integrated effect of components which are affected by the engineering change. It is challenging to detect

and evaluate due to product complexity, different information resources and distinct effects on the life cycle. Some information systems now support prompt response to engineering changes by providing interoperable communication between stakeholders. However, evaluating engineering changes remains a bottleneck preventing prompt response to engineering changes. This is also a challenging problem when applying environmental analysis on ECs. For this reason, we propose a similarity metric of ECs to determine whether ECs are environmentally and semantically similar before proceeding with detailed analysis for knowledge-based carbon footprint estimation. The proposed method is to analyze representation of engineering changes and to define similar changes that are likely to have similar impacts, particularly within the same manufacturing enterprise. For example, if a change is made to a molded cover of one cell phone, it is likely to have impacts similar to those of a change to the molded cover of another cell phone model. The method leads to fast prediction of carbon footprint. The detailed explanation is shown in Section IV.

1.4.2 Carbon footprint estimation with incomplete information

Traditional carbon footprint calculation tools can be verified only if proper information about product and process is available. In the early phase of product design and EC process, predicting accurate environmental impact is not possible because designers cannot provide all related product information due to the degree of freedom in design decision-making. To estimate carbon footprint using incomplete activity data, a knowledge-based approach is utilized. The knowledge-based prediction of

new products and proposed ECs needs to deal with uncertainty of product realization on design parameters. In addition, another issue involves developing the method to enable fast prediction of carbon footprint. To adapt incomplete information with similar cases, incomplete activity data should be estimated by the reuse of the similar products and engineering changes. To develop carbon footprint classification using previously evaluated carbon footprints, carbon calculation is estimated as a boosted linear combination of key activity parameters which are the available information stemming from new product design and ECs.

1.5 Research objective

The goal of this research is to identify the significance of carbon footprint in new product and engineering change using previous carbon footprint data.

The research consists of three categories:

1. **Develop consistent product functions to retrieval functionally similar products**

The task is to development formal representation for product function semantics using web based ontology and rule language. The model facilitates functional design of products in the early design phase and semantic retrieval of the functional model.

2. **Define the classification of ECs in order to determine important effects**

- a) Similarity measure for engineering change: No formal proximity function between engineering changes has defined. This task is to propose an alignment-matching similarity to engineering changes in terms of affected parts. The evaluation between carbon footprint and similarity measure is performed to evaluate the degree of correlation between them.
- b) Clustering method for engineering change: Engineering change clustering method is presented to retrieve effectively similar ECs from a large database.

3. Provide approximate environmental measure using previously evaluated LCA data

- a) Estimating key environmental parameters: A framework is presented to estimate key attributes that can be predicted using previous similar cases and available product information in the early design phase.
- b) Carbon footprint classification using boosted linear approximation: Knowledge-based carbon footprint estimation to predict the significance of carbon footprint with incomplete product information.

The next section presents the outline of the dissertation.

1.6 Outline of the dissertation

This chapter explains a method to enable fast prediction of knowledge-based carbon footprint for new product designs and engineering changes. Chapter II discusses the definition of product functions. It mainly will be about formal representation

of functions for interoperability and shared understanding. Chapter III explains the similarity measure of engineering changes based on the study of engineering change representation and clustering methodology. In chapter IV, we present the method to enable fast prediction of knowledge-based carbon footprint to new product design and engineering changes. Finally, chapter V summarizes research tasks and identifies research contributions and future work.

CHAPTER II

Literature review

This chapter discusses related work in the field of semantic representation of functions, similarity measure of engineering changes, and approximate environmental impact prediction tools. The following sections will discuss specific literature for the three research topics explained in the previous chapter.

2.1 Representation of function

Several researchers have focused on understanding the meaning of a function. The focus of our work is on developing a web-compliant representation that captures appropriate semantics; it is not on providing new definitions in the area of function modeling. The reader is encouraged to refer to works by Erden et al. [4] and Chandrasekaran [5] that provide detailed treatises on several efforts in defining functions.

Umeda et al. [6] define function as “a description of behavior recognized by a human through abstraction in order to utilize it.” They represent function as a tuple capturing both the purpose (function) and physical semantics (behavior). Pahl et al. [7] have proposed one of the most widely used frameworks in which a function is

composed of an input or output flow of material, energy, and signal. They model the overall function and decompose it into sub-functions maintaining consistency of flows throughout the system. However, their approach does not deal with formality of functional representation for automated reasoning and knowledge sharing.

Two independent research efforts with different goals initiated the development of a widely referred functional representation in the mechanical design space. Szykman et al. [8] document the effort within the National Institute of Standards and Technology (NIST) to generate a large taxonomy of functions and flows (over 130 functions and 100 flows) to provide a generic infrastructure and a schema to facilitate the capture and exchange of function information among researchers within a larger design repository project. Stone and Wood [9] proposed the Functional Basis with the overall aim of describing and comparing products functionally. After multiple research and industrial studies of several products, their studies characterizes a mechanical product using the verb-object (function-flow) format. Hirtz et al. [10] present a reconciliation of the two efforts to formulate the current version of the Functional Basis that is intended to comprehensively describe the mechanical design space.

Using the Functional Basis, Kitamura et al. [11, 12] have proposed an ontology of functions to derive functional models from an assembly of parts. The functional ontology contains formal functions and meta-function for semantic representation. Kitamura et al. [13] discuss the development of reference ontology for functional knowledge interoperability.

2.2 Similarity measure for engineering changes

In this section, we present related work only in the context of determining similarity of Engineering Changes. There are very few efforts in this direction. In [14], every EC is composed of distinct product, component, problem types, solutions, and process representations. Similarities are measured in each representation type and the results are linearly combined to obtain the overall similarity between change instances. In [15], past changes that are similar to a proposed change are retrieved based on the similarity of a few specific attribute values. The similarity between values of attribute affected parts is computed using a binary vector-based method. The Issue Based Information System (IBIS) is utilized to determine the similarity between values of reason for change, and a predefined look-up table is employed to determine the similarity between all remaining attributes. These methods are designed to work on a set of pre-identified attributes, are not in the context of predicting the impact on carbon footprint, and implicitly assume that the data structure is not hierarchical.

In the domain of similarity measurements, most methods involves some variations of two main approaches: the geometric models (metric space) approach [16], and the set theoretic approach [17]. The metric space approach represents similarity as a geometric model consisting of objects as points, based on the attribute values, in the multi-dimensional metric space; similarity between two objects is inversely proportional to the distance between the objects. The set-theoretic approach represents objects as a collection of features/attributes, and similarity is expressed in the form

of a linear combination of the measure of common and distinctive features. However, these approaches work on representations that are not hierarchically structured like Engineering Change data.

Similarity, Interactive Action and Mapping (SIAM) [18] is an approach used in psychology to address the problem of structural alignment. In this research, we shall use this approach to compute the similarity between two ECs. More details on the method are provided in the next section.

2.3 Carbon estimation in the early phase of design

Publicly available specification (PAS) 2050 has been developed as an international standard for carbon footprint calculation by the British Standard Institute (BSI). PAS 2050 provides the lifecycle-wide view of all materials, energy and waste flows across all activities in a product's lifecycle. Though the calculation itself is simply calculated by multiplying the activity data by the appropriate emission factors, it is costly and time-consuming due to the need for detailed product information and expertise. This makes PAS 2050 a unsuitable method for the early design process.

In order to overcome PAS 2050's limited application to design process, different simplified methods have been proposed. Graedel et al. [19] proposed a qualitative approach using a checklist and qualitative matrix respectively. These methods can provide the basic information about tradeoffs and environmental awareness to designers. However, these methods are limited when estimating the difference of carbon footprint for existing products in the early design process. Kalyanasundaram

et al. [20] proposed a simplified tool to estimate environmental impact of products quickly using CAD information. Their tools utilize physical information about parts from CAD data, such as mass and material. This information is used to obtain energy consumption and CO_2 generation during manufacturing and transportation. It can be assumed the information about geometry and materials can trigger fast prediction of a product's environmental impact. The learning technique introduced by Sousa et al. [21] estimates lifecycle energy of a product, mostly based on material production energy and use energy. This method utilized the previous LCA data and inventory information related to environmental impact estimations for training. The study demonstrated accuracy and precision in predicting relative differences, and provided an ability to generalize relative difference. However, their study has a problem developing a train model if the resources are distinct.

2.4 Summary

This chapter presented related work for each of three research tasks discussed in this dissertation. First, it introduced the efforts to create function definitions and representations in the area of function modeling. Second, it presented the similarity measure concept in the EC domain and provided general methodologies used in defining similarity measure. Last, this section showed several approaches to enable awareness of product carbon footprint in the early design phase.

CHAPTER III

Function representation for understanding product semantics

3.1 Motivation

As mentioned in Section I, environmental impact analysis in the design phase aims to estimate quantitatively the environmental impact of alternative concepts as well as of design decisions for sustainable product development. For the problem of incomplete environmental information, environmental impacts are measured by previously evaluated LCA of similar products. However, there exist few frameworks and tools to express the identity of product functions, and shared understanding of product system is also lacking. In this chapter we propose functional semantic representation to understand the functional identity of product that will provide product information about similar products. Section 2.4 describes functional semantic representation and expressiveness of product functions with respect to functional decomposition. Section 2.5 shows the validation of semantic queries to retrieve functionally similar product.

3.2 Objective

Designers need to recognize the overall evaluation for decision and selection of alternatives, because most environmental impacts are decided in the design phase. When redesigning a previous product, we can reuse product knowledge for an environmental impact measure. This assumes that functionally similar products need to be considered with respect to product design, manufacturing data, use profile and recyclability. This approach requires the formal representation of product functions to share functional knowledge. At this point, we can define the problem as follows:

For: New product function information and database of old products

Find: Functionally similar products

3.3 Modeling the Function Semantics Representation (FSR)

An ontology is an explicit specification of a conceptualization [22]. It usually introduces concepts, properties of concepts, relationships between the concepts and other additional constraints in a machine-readable format. In the Function Semantics Representation (FSR), the core ontology is encoded using Web Ontology Language (OWL). Rules encoded using the SWRL are used to explicitly capture advanced semantics essential for a usable representation of product functions. The following sections explain the core ontology, the need to use SWRL, and the corresponding rule extensions.

3.3.1 The core OWL ontology

The core concepts and taxonomies which are encoded using OWL in the FSR are based on the Functional Basis, proposed by Hirtz et al. [10], which defines a Function as a concept involving the transfer or transformation of an input Flow into an output Flow. The intent of this paper is not to present a detailed taxonomy; only representative descriptions are discussed.

Core concepts

The concept *Flow* has three distinct subconcepts-*EnergyFlow*, *MaterialFlow* and *SignalFlow*-which represent the flows of energy, material and signal respectively. Flows of these three elements are treated as basic flows, and they are classified into subconcepts, such as *ElectricCurrentFlow*, *LiquidFlow*, and *VisualSignal*. These definitions of Flow and the corresponding taxonomy are as described in [4]. Flows are represented by a set of distinct concepts in the ontology by explicitly defining that all *Flow* concepts are disjoint from one another. For example, in the RDF/XML syntax for OWL, Figure 3.1 states that the *EnergyFlow*, which is a type of *Flow*, is disjoint from *SignalFlow* and *MaterialFlow*.

Every flow or medium has some time-dependent property such as location, mass, and composition. Therefore, all instances of flows are different. Thus, even if the same molecule of water is transmitted through a pipe, the input water flow is different from the output water flow, since the location of the water molecule changes at


```

<owl:Class rdf:ID="EnergyFlow">
  <rdfs:subClassOf rdf:resource="#Flow"/>
  <owl:disjointWith rdf:resource="#SignalFlow"/>
  <owl:disjointWith rdf:resource="#MaterialFlow"/>
</owl:Class>

```

Figure 3.1: Representing disjoint Flows in RDF/XML. Bold terms are important elements in the description

```

<owl:AllDifferent>
  <owl:distinctMembers rdf:parseType="Collection">
    <rdf:Description rdf:about="#Heat"/>
    <rdf:Description rdf:about="#Force"/>
    ...
    <rdf:Description rdf:about="#Decibel"/>
  </owl:distinctMembers>
</owl:AllDifferent>

```

Figure 3.2: Every individual flow is different from every other flow.

the different time instants. This is captured through the owl:AllDifferent property applied to all individuals as shown in Figure 3.2.

Following are some of the important properties associated with the concepts Function and Flow that enable us to capture the definition of a product function, as stated earlier. More details are out of scope of this paper:

1. A *Function* may have an input flow. This relation is captured by the property, *hasInputFlow*, which has an inverse property called *isInputFlowOf*. In the Description Logic Syntax [23], this is represented as: $hasInputFlow \equiv (isInputFlowOf)^-$.
2. Similarly, a *Function* may have an output flow, which is captured by the property, *hasOutputFlow*.
3. A *Function* has exactly one name. This relationship is captured by the datatype property, *hasName*, with string as its range. It is used as an identifier for the

```

<owl:Class rdf:ID="Function">
  ...
  <owl:unionOf rdf:parseType="Collection">
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasInputFlow"/>
      <owl:someValuesFrom rdf:resource="#Flow"/>
    </owl:Restriction>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasOutputFlow"/>
      <owl:someValuesFrom rdf:resource="#Flow"/>
    </owl:Restriction>
  ...
</owl:Class>

```

Figure 3.3: Partial RDF/XML code for definition of Function. Some terms are in bold to highlight important elements in the description

individual instance of a *Function*.

Using these properties, a Function is defined as an OWL class that has exactly one name and has an output flow or an input flow (or both). In RDF/XML syntax, its partial expression is as shown in Figure 3.3.

In Description Logic syntax, this partial definition is expressed as follows:

$$Function \subset (\exists hasInputFlow.Flow \wedge \exists hasOutputFlow.Flow) \wedge = 1 hasName$$

The concept *Function* has eight distinct subconcepts (Figure 3.4): *Branch*, *Channel*, *Connect*, *Convert*, *ControlMagnitude*, *Provision*, *Support*, and *Signal*. Each subconcept may have further subconcepts. For example, the function *Channel* has four subfunctions: *Export*, *Guide*, *Transfer* and *Import*. The function *Transfer* has two subfunctions: *Transmit*, and *Transport*. Each of the leaf concepts is expected to be a unique function defined by certain interrelationships between the input and output flows.

Thus, multiple properties, such as *hasName*, *hasInputFlow* and *hasOutputFlow*

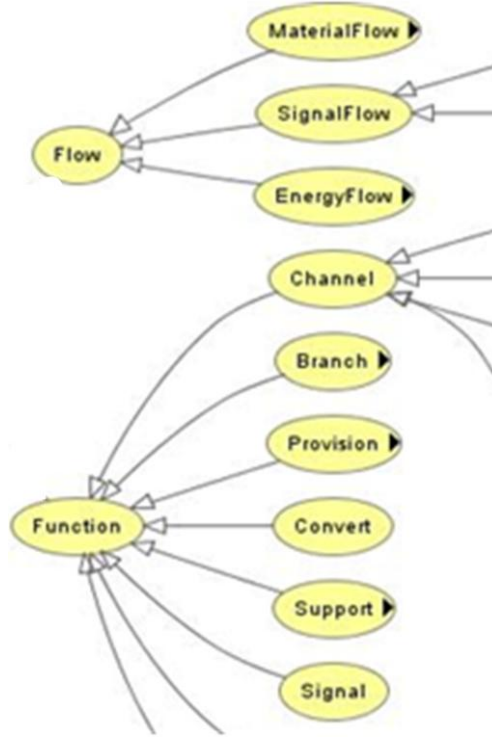


Figure 3.4: Partial taxonomy of the FSR. The small triangles indicate corresponding subconcepts not shown in the figure

are defined using OWL in the FSR to capture the relationships between *Function* and *Flow*. OWL provides facilities to impose cardinality, quantitative or value restrictions on these properties. Using these basic concepts and properties, more usable concepts are defined. For example, the function *Branch* is a function that has exactly one input *Flow* and more than one output *Flow*. This is defined as:

$$Branch \subset Function \wedge \forall hasInputFlow.(MaterialFlow \wedge EnergyFlow)$$

$$\wedge = 1hasInputFlow \wedge \geq 2hasOutputFlow$$

Important properties in the FSR

The previous section discussed the taxonomic relations between functions and flows based on the *Functional Basis*. However, using only the number and types

Table 3.1: Number of input and output flows may not be always used to differentiate functions from one another. The meaning presented by the original textual definitions from the Functional Basis [9] needs to be encapsulated

Function	Number of flows	Definition in <i>Functional Basis</i>
<i>Import</i>	0 input flow	To bring in a flow (material, energy, signal) from outside the system boundary
	1 output flow	
<i>Supply</i>	0 input flow	To provide a flow from storage
	1 output flow	
<i>Export</i>	1 input flow	To send a flow(material, energy, signal)
	0 output flow	
<i>Store</i>	1 input flow	To accumulate a flow
	0 output flow	

is not enough to logically differentiate functions from each other. For example, as shown in Table 1, *Import* is a subfunction of the function *Channel* and has exactly 0 input and 1 output flows. Please note that while this might seem counterintuitive, the idea is to capture the flows within the system boundary; the input flow to *Import* is outside the system boundary. Another function, *Supply*, which is a subfunction of *Provision* has the same configuration, i.e., 0 input flow and 1 output flow. Similar ambiguities are presented due to similarities in the number of input and output flows for *Export* and *Store* functions. The *Functional Basis* defines these functions using textual descriptions as shown in Table 1. It is important to capture these explicitly.

In order to achieve meaningful semantics, two new physical state identifiers, *hasMagnitude* and *hasDOF*, outside of the original schema of the Functional Basis [9], are added to the ontology.

The property *hasMagnitude* with domain of both *Function* and *Flow* concepts is introduced as shown in Figure 3.5. For simplicity of implementation, units of the magnitude are not considered to be significant and the range is float. It captures

```

<owl:DatatypeProperty rdf:ID="hasMagnitude">
  <rdfs:domain>
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#Flow"/>
        <owl:Class rdf:about="#Function"/>
      </owl:unionOf>
    </owl:Class>
  </rdfs:domain>
  <rdfs:range rdf:resource="&xsd;float"/>
</owl:DatatypeProperty>

```

Figure 3.5: Definition of `hasMagnitude` in OWL. Some terms are in bold to highlight important elements in the definition

values of physical properties, such as temperature, pressure and force of the relevant flow. When a *Function* is its domain, *hasMagnitude* represents the maximum amount of allowable flow through the system. Thus, for *Import*, the value that *hasMagnitude* takes is very large (theoretically infinite), while for *Store* it takes a finite predefined value. These can be used to distinguish the functions logically.

The property *hasDOF* is introduced to represent the Degree Of Freedom (DOF) of the flows and is associated with the functions *Position*, *Secure* and *Stabilize* which have implicit assumptions about the associated DOFs. It just defines the extent of free motion with the number and does not capture geometric transformations.

3.3.2 SWRL rules in the FSR

After studying the literature and several examples, we identify the following abilities that a representation must have to capture sufficient semantics about product functions.

1. Counting the number of flows.

2. Identifying the type of a flow.
3. Determining same types of flows, i.e., homogeneity of flows.
4. Determining different types of flows, i.e., heterogeneity across flows.
5. Comparing values of different properties.

We observe that OWL is not expressive enough to provide the above abilities that are essential to encapsulate the semantics of product functions. SWRL, which is a rule-layer on top of OWL, provides these facilities and is used in our work. The reader is referred to a more detailed treatise on SWRL [24] to understand several concepts, such as *abox* and *tbox* built-ins.

The difference between the abilities to determine the same type of flows (3 above) and determine different types of flows (4 above) should be noted. These could have been considered as complementary abilities. However, OWL supports open-world assumption, i.e., absence of the truth of a statement does not necessarily imply that it is false. In addition, SWRL does not support the representation of negation. Therefore, there is a need to identify each of them separately. Using representative examples, the following explains the utility of SWRL in the above operations.

Counting the number of flows

As discussed earlier, the function *Import* has no input flow and is a subclass of the function *Channel*. The challenge is in defining that the function has no input flow. One option is to use a *hasProperty* method that returns true when the individual has that property. Thus, if an individual *Function* does not have a property

$ \begin{aligned} & Channel(?x) \\ & \wedge abox:hasNumberOfPropertyValues \\ & \quad (?y, ?x, hasInputFlow) \\ & \wedge swrlb:equal(?y, 0) \\ & \rightarrow Import(?x) \end{aligned} $

Figure 3.6: Counting the number of properties using `abox:hasNumberOfPropertyValues` for the function *Import*

hasInputFlow, then it is of the type *Import*. However, this requires using a negation in the SWRL to express "no input flow." This is not possible in SWRL due to the Open World Assumption.

In order to count the number of properties and therefore the number of flows as shown in Figure 3.6, the built-in `abox:hasNumberOfPropertyValues(?a, ?b, ?c)` is used to determine the number of times *a* that a certain property *c* is used in a certain individual *b*. Thus, if an instance *x* of the function *Channel* has *y = 0* number of properties *hasInputFlow*, then *x* is of the type *Import*. The partial Resource Description Framework (RDF)/ eXtensible Markup Language (XML) syntax for this rule is shown in Figure 3.7. The remainder of this chapter uses the human readable syntax as shown in Figure 3.6.

Identifying the type of a flow

SWRL can be used to identify the specific type of flow associated with an instance of a function. As defined by [10] in the *Functional Basis*, *Signal* function "provides information on a material, energy or signal flow as an output signal flow. The information providing flow passes through the function unchanged." This rule is represented in Figure 3.8, which says that an instance *x* of any *Function* that has

```

<swrl:Imp rdf:ID="Rule_for_channel_import">
  <swrl:head>
    <swrl:AtomList>
      ...
      <swrl:argument1 rdf:resource="#x1"/>
      <swrl:classPredicate rdf:resource="#Import"/>
    </swrl:ClassAtom>
    <swrl:builtin rdf:resource="http://www.w3.org/2003/11/swrlb#equal"/>
    </swrl:BuiltinAtom>
    ...
    <swrl:BuiltinAtom>
      <swrl:builtinrdf:resource="http://swrl.stanford.edu/ontologies/built-
        ns/3.3/abox.owl#hasNumberOfPropertyValues"/>
      <swrl:arguments>
        <rdf:List>
          <rdf:first rdf:resource="#y"/>
          ...
          <rdf:first rdf:resource="#hasInputFlow"/>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
        </rdf:List>
        </rdf:rest>
        <rdf:first rdf:resource="#x1"/>
        ...
      </swrl:classPredicate rdf:resource="#Channel"/>
      ...
    </swrl:arguments>
  </swrl:head>
  ...
</swrl:Imp>

```

Figure 3.7: Partial RDF/XML syntax showing SWRL rule for the function *Import*. Corresponding human readable syntax is shown in Figure 3.6

```

Function(?x)
  ∧ abox:hasNumberOfPropertyValues(1, ?x, hasInputFlow)
  ∧ abox:hasNumberOfPropertyValues(2, ?x, hasOutputFlow)
  ∧ hasOutputFlow(?x, ?x2)
  ∧ abox:hasClass(?x2, StatusSignal)
  → Signal(?x)

```

Figure 3.8: Using *abox:hasClass* to determine type of *Flow* in identification of the function *Signal*

one input flow and has two output flows and has *StatusSignal* as one of the output flows is of the type *Signal*. The built-in *abox:hasClass(?a, ?b)* is used to assert if individual *a* is an instance of the class, *b*.

Determining same types of flows

The function *Branch* means “to cause a flow (material, energy, signal) to no longer be joined or mixed” [9]. It has two sub-functions: *Distribute* and *Separate*. *Distribute*

$ \begin{aligned} & Branch(?x) \\ & \wedge abox:hasNumberOfPropertyValues(?y, ?x, hasInputFlow) \\ & \wedge abox:hasNumberOfPropertyValues(?z, ?x, hasOutputFlow) \\ & \wedge swrlb:lessThan(?y, 2) \\ & \wedge swrlb:greaterThan(?z, 1) \wedge hasInputFlow(?x, ?x1) \\ & \wedge hasOutputFlow(?x, ?x2) \\ & \wedge abox:hasClass(?x1, ?class1) \wedge abox:hasClass(?x2, ?class2) \\ & \wedge tbox:sameAs(?class1, ?class2) \\ & \rightarrow Distribute(?x) \end{aligned} $

Figure 3.9: Using *tbox:sameAs* to detect the same types of flows in the function *Distribute*

breaks up a single *Flow* so that the output flows and the undistributed input flow are similar. *Separate* breaks a flow into distinct flows. However, the FSR includes the restriction that all flows are different from one another (See Figure 3.2). In this particular case, the function *Branch* implicitly assumes the same type of flows. To implement this definition, SWRL built-in *abox:hasClass* is used to assert the class of the individual under consideration. The built-in *tbox:sameAs* is used to determine if the two flows are the same. This is implemented as shown in Figure 3.9 and indicates that if an instance of the function *Branch* has $y < 2$ input flows and $z > 1$ output flows, and the types of the input ($x1$) and output flows ($x2$) are the same, then it is an instance of *Distribute*.

Determining different types of flows

The function *Convert* is used to indicate the change from one form of a flow (material, energy, signal) to another. Accurately defining this function requires determining that the type of input flow is different from the type of output flow. This is not possible using OWL. As shown in Figure 3.10, the built-in *tbox:differentFrom* is used for this purpose. Thus, if an individual x has an input flow $x1$ and an output flow $x2$, such that $x1$ is different from $x2$ and the number of input and output flows

```

Function(?x)
∧ hasInputFlow(?x, ?x1) ∧ hasOutputFlow(?x, ?x2)
∧ abox:hasClass(?x1, ?class1) ∧ abox:hasClass(?x2, ?class2)
∧ tbox:differentFrom(?class1, ?class2)
∧ abox:hasNumberOfPropertyValues(?y, ?x, hasInputFlow)
∧ abox:hasNumberOfPropertyValues(?z, ?x, hasOutputFlow)
∧ swrlb:equal(?y, 1) ∧ swrlb:equal(?z, 1)
→ Convert(?x)

```

Figure 3.10: Using *tbox:differentFrom* to detect the different types of flows in the function *Convert*

```

Support(?x) ∧ hasInputFlow(?x, ?y1)
∧ hasDOF(?y1, ?y2) ∧ hasOutputFlow(?x, ?z1)
∧ hasDOF(?z1, ?z2) ∧ swrlb:lessThan(?z2, ?y2)
∧ swrlb:equal(?z2, 0)
→ Position(?x)

```

Figure 3.11: Using SWRL built-ins to compare values of different properties and detect the function *Position*

are both 1, then x is an instance of the function *Convert*.

Comparing values of different properties

The function *Position* can be defined as a type of the function *Support*, such that the degree of freedom $z2$ of the output flow is less than the degree of freedom $y2$ of the input flow, and that $z2=0$. It might seem that the restriction that $z2 \leq y2$ is redundant. However, such clear axioms are added to prevent the ontology and the reasoner from instantiating and validating invalid concepts. Figure 3.11 shows the use of *built-ins swrlb:LessThan* and *swrlb:equal* to compare the values of the property *hasDOF*.

3.4 Case Studies

In this research, we have used the Protégé platform [25] to model the Function Semantics Representation (FSR) including the SWRL rules and the example func-

tional models. The Pellet reasoner [26] and Jess rule engine [27] are used to reason with the ontology and the example functional models.

In this section, we discuss two examples:

1. To demonstrate the ability to check the consistency of a functional model, and
2. To demonstrate semantic retrieval of functionally similar products.

3.4.1 Consistency Checking

Using formal semantics of the functions, the logic-based FSR enables a framework to ensure/validate the consistency of the functional model of a manufactured product. A model is inconsistent or incorrect if any assertion derived from the model violates any of the facts or constraints built into the FSR. Consider the ABS brake system modeled in the Design Repository at Missouri University of Science and Technology [28]. In particular, we focus on the component *Relay*, shown in Figure 3.12, which enables the function *Actuate* with input flow *Control Signal*, and output flow *Electrical Energy*.

The *Actuate* function is defined in the functional basis as “to commence the flow of energy, signal, or material in response to an imported control signal.” In the literal sense, it seems correct that there should be one input signal flow and one output energy flow as shown in Figure 3.12. However, the function *Convert* has been defined as “to change from one form of a flow (material, energy, signal) to another.” Thus, there is the chance that a logical reasoner will be unable to differentiate between *Actuate* and *Convert*, since enough semantics are not captured.

We modeled this into the FSR and applied the Pellet DL reasoner and Jess rule

System: brake system					
Artifact Name	relay		Artifact Photo		
Sub Artifact Of	Not Specified		 click on image for full size		
Quantity	1				
Description					
Artifact Color(s)	Not Specified				
Component Naming	electric switch				
Input Artifact	Input Flow	Subfunction	Output Flow	Active Flow	Output Artifact
computer	control	actuate	electrical	Active	abs pump
Supporting Functions					
There are no supporting functions defined for this artifact.					

Figure 3.12: Partial functional model of a *Relay* used as a part of an ABS Brake System. Picture and data taken from [28]

engine to evaluate the consistency of this model. The Pellet DL reasoner enables validations of assertions based on the OWL axioms. However, this first requires classification of the different individuals (e.g., relay) used in the functional model into appropriate classes (e.g., *Convert* or *Actuate*). Therefore, the Jess rule engine is used to develop a classification based on the rules first and then the Pellet reasoner is used to check its consistency. Figure 3.13 shows a screen grab of the result of the consistency checking within the Protégé environment. The relay is classified into *Convert* and the original *Actuate*. However, an instance cannot be classified into two different functions, since the functions are defined as mutually exclusive. The Pellet reasoner shows that there is an error in the consistency of the functional model and identifies that the component *Relay* was forced to belong to the function “*Convert*” and its complement at the same time.

The designer can use this knowledge of inconsistency to correct the original functional model. In this case, realistically (and as captured in the FSR) the function *Actuate* must have some input energy, since the act of “commencing” an energy flow

requires some input energy flowing into the function. Furthermore, output energy without any input will violate the law of conservation of energy. Therefore, assuming that “*Actuate*” was indeed the correct function, the designer needs to indicate an input energy flowing into the relay. This may require modification of other functions captured in the representation of the ABS brake system.

3.4.2 Semantic retrieval

The support for logical reasoning and inference coupled with semantic definitions of the functions enables semantic retrieval of functionally similar products. The FSR captures the semantics of functions through the relations between individual functions and flows.

We use a simple database to demonstrate semantic retrieval based on FSR. Our data repository consists of 30 products that are modeled using the Design Repository at Missouri University of Science and Technology [28]. The product types that were selected are as follows: coffee maker, power screw driver, jigsaw, hand vacuum, toothbrush, drill, iron, sander, cordless water kettle, ABS Brake system, dust buster, water pump, circular saw, and can opener. Multiple individuals of these product types were created. Consider a hypothetical water kettle with a simple partial functional model as shown in Figure 3.14. The product consists of the functions *Convert*, *Change*, and *Store*. The two primary flows are *EnergyFlow* and *MaterialFlow*. The goal is to determine those products that are functionally similar to the water kettle.

It should be noted that SWRL is not a query language and does not have the

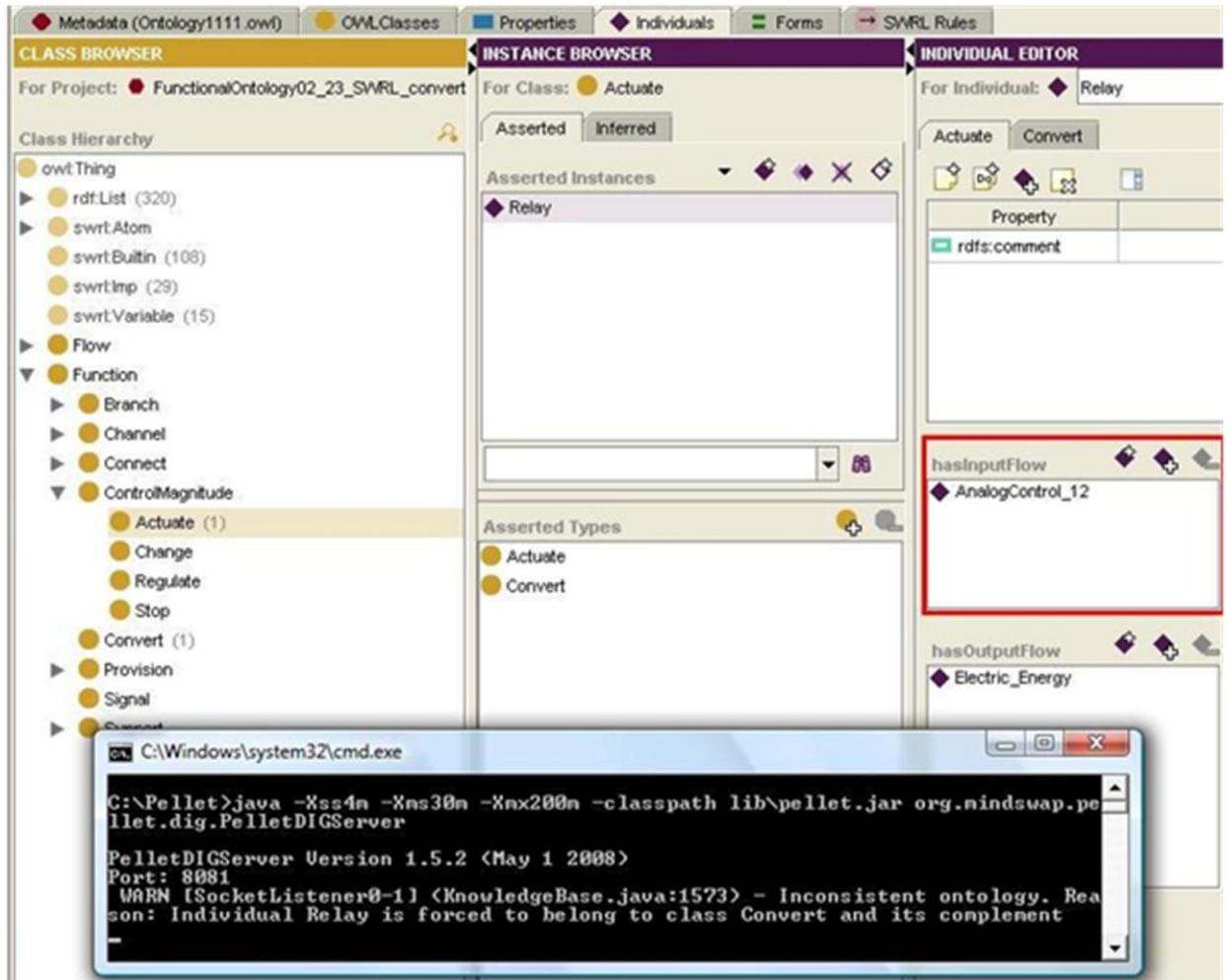


Figure 3.13: Result of consistency checking indicates that the function, *Actuate* of the Relay in Figure 3.12 is invalid. The Jess rule-engine forces it to be of the two types *Convert* and *Actuate* simultaneously, and the Pellet reasoner indicates that the model is inconsistent

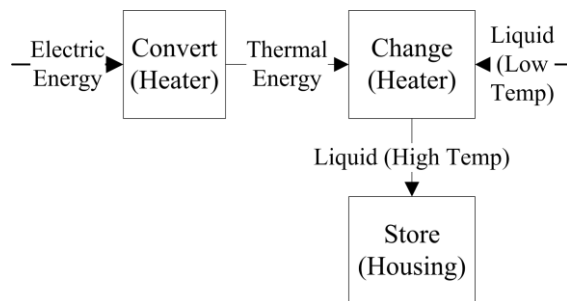


Figure 3.14: Partial functional model of a hypothetical water kettle used to demonstrate semantic retrieval of functionally similar products

```

Convert(?x) ∧ hasOutputFlow(?x, ?output_1)
∧ ThermalEnergy(?output_1)
∧ hasInputFlow(?z, ?output_1)
∧ Change(?z) ∧ hasOutput(?z, ?output_2)
∧ Liquid(?output_2) ∧ hasInputFlow(?b, ?output_2)
∧ Store(?b) ∧ hasName(?x, ?name)
→ sqwrl:select(?name)

```

Figure 3.15: SQWRL query to search a product containing *Convert*, *Change*, and *Store* function (so that it is similar to our hypothetical water kettle)

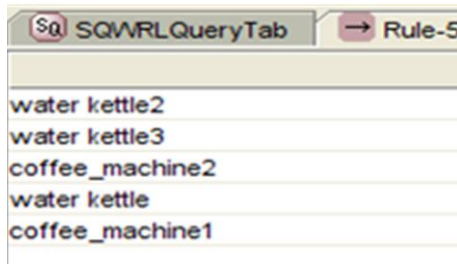


Figure 3.16: Functionally similar instances retrieved into the SQWRL Query Tab in Protégé

ability to support knowledge extraction, i.e., extracting information from the FSR and the functional models, or the ability to reason with the information contained in it. We employ a query language called SQWRL [29] which extends SWRL to enable efficient querying. The corresponding partial SQWRL query for the example is shown in Figure 3.15. It asks to retrieve that product which has a function of type *Convert*, with an output flow of *ThermalEnergy*, such that this output of *ThermalEnergy* is an input to a *Change* function that outputs a *Liquid* which in turn is input to a function, *Store*. This query is consistent with the model shown in Figure 3.14.

The results of the retrieval are shown in Figure 3.16. Products of the type Cordless Water Kettle and Coffee Maker are found to have the similar functions.

Currently, the procedure uses only the information about functions to retrieve

similar products. There is a need to use other information, such as product size, shape, etc. to enable effective searches. At the same time, using the ability of SWRL to compare the magnitudes of the flows can be useful, for example, to distinguish between a water kettle and a steam engine.

3.5 Summary

Defining or understanding a product in terms of its functions facilitates a wide variety of tasks such as design synthesis, modeling, and analysis. However, the lack of a formal representation for these elements of product model data creates a barrier to their effective capture, exchange and reuse. Increasingly, information resources in PLM are expected to be available through Web services in the Semantic Web. Ontologies are a part of the set of Semantic Web technologies used to represent semantics and promote integrated and consistent access to data and services.

This chapter presented the development of a Function Semantics Representation (FSR) as a fundamental characterization of product functions for use in Product Lifecycle Management (PLM). New properties and rules are defined using OWL and SWRL to accurately capture the semantics beyond the existing schema-based approaches. Application to consistency checking of a functional model and semantic retrieval of functionally similar products are demonstrated.

The FSR can be used to automate several applications, such as defect detection

in parts, automated rule-based development of the functional model, and searching for parts that satisfy a functional requirement. Consistency checking could be used to develop productivity interfaces that guide the designer in developing a correct functional model, for example, by showing missing flows associated with a particular function. In addition, it is expected that SWRL can communicate with traditional relational databases. This feature is very attractive since most information in the product development world is still stored in relational databases that may not be easily replaceable in the near future.

In Section V, an approach using the FSR has been proposed to obtain functionally similar products to enable knowledge-based evaluation about the sustainability of a manufactured product, since lifecycle analysis and some other sustainability evaluation techniques start by considering the functional decomposition of the product.

CHAPTER IV

Clustering Engineering Changes

Evaluating the carbon footprint of engineering changes refers to assessing the carbon emission by component changes involved in a proposed engineering change. Evaluating carbon emission of component changes increases EC process time and cost for considering additional change information related to carbon emission over lifecycle. The objective of this research is to provide Engineering Change (EC) clusters that enable fast response to engineering change evaluation with minimized cost. EC clusters can provide the environmentally important feature of predicting carbon footprint for a proposed EC that just contains virtual changes to related products or processes. In this section, we propose a novel clustering method to classify ECs that embed similar carbon emission through defining a similarity measure for engineering changes and a scalable clustering methodology applicable for parallel computing.

4.1 Motivation

An Engineering Change (EC) refers to any change to shape, dimensions, material, etc. of a part or assembly after the initial design has been released [15] and

has been considered inevitable for successful product development [30]. EC needs to make a decision based on negotiation between stakeholders' needs [31]. There has, however, been lack of research on the environmental impact of ECs. To include environmental analysis in the evaluation of ECs requires all data related to environmental parameters. However, the impact of ECs is so interdependent that it cannot be predicted by a single coordinator. It requires that related stakeholders know the environmental parameters and provide them to environmental experts. It can be possible to develop ECM that allows immediate communication to share EC information in parallel. However, it is still a time consuming and costly process, because evaluating environmental impact of ECs must be derived from the change of environmental parameters after stakeholders determine the impact of ECs.

Companies have struggled to process engineering changes in shorter lead time. Globalized business and environmental needs have represented challenges in product development for most companies. A wide variety in customers' needs and shifts in the global market require prompt and flexible responses to product and process change requirements. At this point, we propose a semantic clustering of engineering changes to allow fast response to environmental evaluation.

The proposed method provides EC clusters in order to predict by analogy, comparing environmental parameters with previous evaluations [15]. The EC data for the method is assumed to be sufficiently large to cover all cases of engineering changes as well as organizational knowledge for a specific business. If retrieved ECs do not have similar clusters, the evaluation process will require continuing collaboration with en-

vironmental experts.

4.2 Objective

The objective of the research is to develop a clustering method that recognizes a set of ECs embedding similar carbon emission. To develop the EC clustering method, the proposed method should clarify EC representation, definition of EC proximity, and EC clustering methodology.

1. *EC representation:* Identify the data structure of engineering changes with respect to existing studies and standards. In particular, the information related to carbon emission calculation is specified to define the semantics of proximity.
2. *Definition of EC proximity:* Determine similarity measure between Engineering Changes (ECs) with the goal of using it in assessing the carbon footprint of engineering changes.
3. *EC clustering methodology:* Determine a clustering method applicable for the characteristics of the proposed similarity measure and suitable to the dynamics and complexity of engineering changes.

4.3 Engineering change data structure

EC data structure aims to capture the objects and their change types, such as deviation of objects, reason for changes, and change process. The objects in EC data structure refer to a placeholder for identifying product data, which includes

parts, assemblies, tools and their use.

EC data structure has been developed by individual companies and researchers to satisfy the needs of stakeholders. Pikosz and Malmqvist proposed EC information systems that generally deal with the context of product documentation and product structure [32]. Keller et al introduced the method called CPM (Change Prediction Model), which utilizes component linkage to predict impact propagation [33]. In addition, there is a international standard to represent engineering changes. ISO10303-239 known as Product Life Cycle Support (PLCS) became an international standard in 2005. It defines an information model for managing product change in a product life cycle. The common objective of these EC data structures is to represent related objects and their changes.

In this section, we propose an EC information model that involves the objects of engineering changes and the output objects, as shown in Figure 4.1. The basic model is based on the SASIG representation of an Engineering Change [34], in which every Engineering change has “input affected objects” and “output affected objects”. “Input affected objects” refers to objects that are considered for the change and will be omitted after the EC. “Output affected objects” refers to objects that are added as a result of the EC. Typical affected objects in an EC include product assemblies, components, machine tools, and so on [14]. In particular, we focused on product oriented objects because the carbon footprint calculation excludes immaterial emission sources and human inputs to processes.

In general, engineering change data is highly structured, i.e., objects are linked

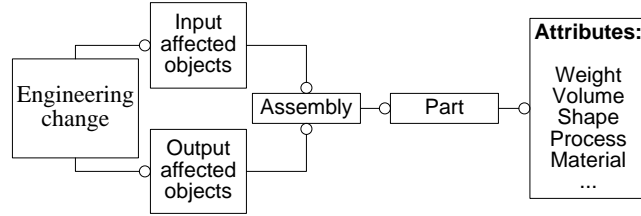


Figure 4.1: Partial schematic of an Engineering Change. Every change has input and output affected objects, each represented as Assembly. An Assembly is made of a vector of parts. Parts have multiple attributes that take a value.

to each other. Object is an entity that cannot take a data value (numeric, text, etc.); its attributes can take data values. For example, an assembly has other parts in it. Furthermore, one product might represent as assembly as a sequence (A, B, C) of three components, while another might represent it as a sequence (B, C, A), and yet another might represent it with just two components (D, A) with D representing a replacement for B+C. Therefore, comparing Engineering Changes involves not just matching attributes, but determining which entities in the data structure align with one another. For example, we may want to find if a change to a sedan with a green door is similar to the one for the green hood of a convertible. However, matching the attribute “color” really does not add value. It is important to evaluate if the “door of a sedan” is aligned to the “hood of a convertible” before we consider matching the color. Therefore, before determining similarity, it is important to align the objects so that they correspond to each other appropriately.

Engineering changes consist of distinct attributes to represent characteristics of change over the life cycle. We have classified three types of attributes: numeric attributes, characteristic attributes, and semantic attributes. The attributes of engineering changes are not clearly defined and may have different impacts on carbon

Table 4.1: Product features over product lifecycle

Lifecycle phase	Product attribute
Design	Component name Weight Volume Color name Lifetime Function Material name Material density Material yield stress Material tensile stress Shape name Thickness
Manufacturing	Mfg process Machine tool name Tool cost Equipment cost Assembly type name Fastener number Kind of fastener Time for assembly Time for disassembly
Product use	Energy consumption Use cost Maintenance cost
Distribution	Freight Transportation Travel distance
Disposal	Disposal strategy name

emission. In this study, the attributes used from the engineering changes are 28 attributes associated with an assembly/parts as shown in Table 4.1. These are selected based on a study of literature reviews and commercial tools over a product lifecycle.

4.4 Similarity measure for engineering changes

As discussed earlier, utilizing past change knowledge to evaluate the impact of a proposed Engineering Changes (ECs) effect requires an approach for computing similarity between ECs. The problem addressed in this paper is:

For: A proposed Engineering Change (EC)

Given: Knowledge-base containing past ECs

Determine: Similarity between the proposed change and each past EC

with the goal of ultimately using this knowledge

in assessing the carbon footprint of the proposed change.

Based on EC data structure defined in Section 4.3, we propose an alignment matching similarity measure. The overall idea of calculating EC similarity is to determine how products configured with similar components are similarly changed. We now will explain the concept of EC similarity to compare input affected objects and output affected objects between two ECs.

4.4.1 EC similarity measure using alignment matching

Let $O_{in,j}$ represent the set of input affected objects and $O_{out,j}$ represent the set of output affected objects for a proposed Engineering change (C_j). Let k suffix represent another EC. Then, the overall similarity between the two engineering changes, C_j and C_k , is calculated as the weighted linear combination of $Sim_{in}(O_{ip,j}, O_{ip,k})$, the similarity between input affected objects C_j and C_k and $Sim_{out}(O_{out,j}, O_{out,k})$,

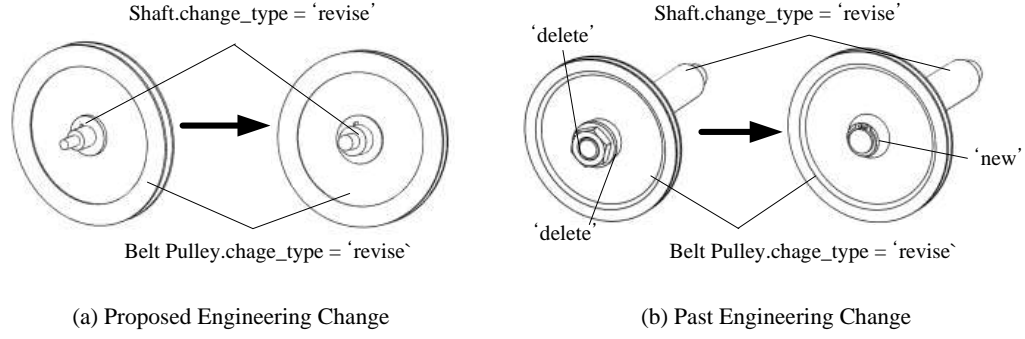


Figure 4.2: Example of engineering changes used to explain the determination of similarity

and the similarity between output affected objects of C_j and C_k , and is given as:

$$Sim(c_j, c_k) = w \times Sim_{in}(O_{ip,j}, O_{ip,k}) + (1 - w) \times Sim_{out}(O_{out,j}, O_{out,k}) \quad (4.1)$$

In this section, we discuss SIAM and its application to obtain the value of $Sim_{in}(O_{ip,j}, O_{ip,k})$ and $Sim_{out}(O_{out,j}, O_{out,k})$. Similarity, Interactive Activation, and Mapping (SIAM) is a quantitative model of similarity developed to capture relational and structural similarity between compared things, in our case, ECs, at the conceptual level, or $O_{ip,j}$ and $O_{ip,k}$ as defined in Equation (4.1). It assumes an interactive process of activations such that correspondences between objects of compared things mutually influence each other. The fundamental aspect is that matching and different attributes have a stronger influence on similarity if they belong to corresponding objects.

Consider the two ECs represented in Figure 4.2. The proposed EC is the change in the diameter of the shaft that is assembled along with a belt pulley. The EC with which we want to find the similarity is about the change in the diameter of a shaft assembled with a chain sprocket. We have selected an example with such

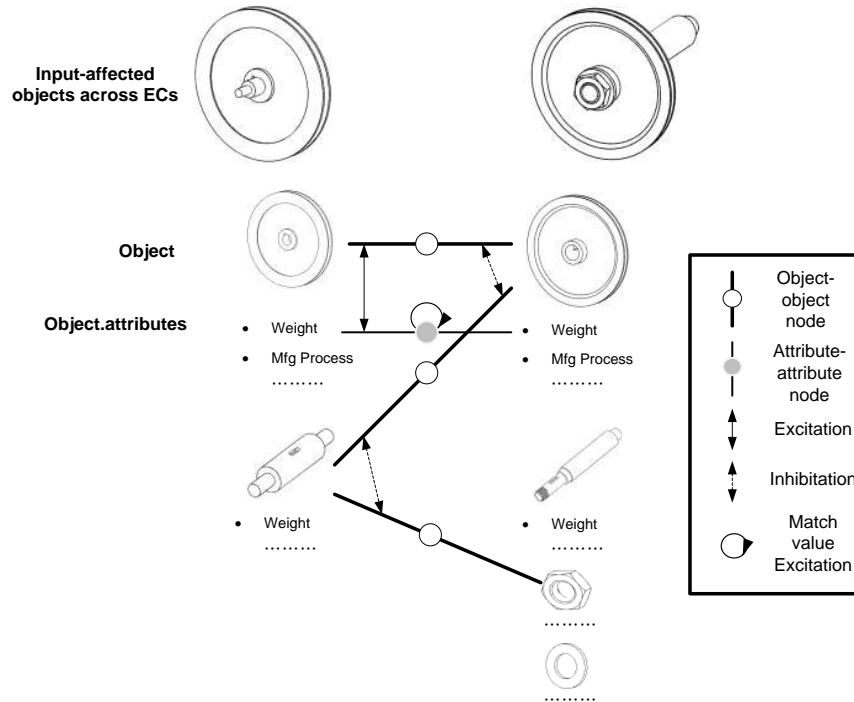


Figure 4.3: Concept used in the calculating similarity ECs

close similarity only for simplicity of explanation. The proposed procedure applies to assemblies with different number of components across the two ECs.

In the following, we explain the overall concepts and procedure used in determining similarity between the input affected objects of the two ECs shown in Figure 4.2.

Concepts used in the procedure

The concepts used in SIAM and applied to EC similarity are explained briefly here. The idea is to obtain the similarity between input affected objects (assembly of small diameter shaft + pulley) of proposed EC with input affected objects (assembly of thread jointed shaft + pulley) of past EC, as shown in Figure 4.2.

The overall architecture is that of a network model consisting of nodes that excite or inhibit each other. A node is defined as a set of two entities in correspon-

dence (shown by circle on thick line in Figure 4.3). We have two different types of nodes: object node and attribute node. Object-Object node represents the hypothesis that two objects are aligned to each other, whereas Attribute-Attribute node implies that two attributes are aligned to each other. For example, in Figure 4.3, Pulley-Pulley form an object node, whereas Pulley.Weight-Pulley.Weight node is an attribute node. As the activation of a node increases, the two entities (objects or attributes) in that node are placed in stronger correspondence. The relative strengths of the correspondences are decided based on structural consistency. Nodes that are consistent with each other excite/support each other. For example, in Figure 4.3, the Pulley.Weight-Pulley.Weight Attribute-Attribute node supports the similarity of the Pulley-Pulley Object-Object node; i.e., they are consistent and therefore activate each other (represented by solid line with two sided arrows). On the other hand, nodes that are inconsistent with each other inhibit each other. For example, in Figure 4.3, the Pulley.Weight-Shaft.Weight Attribute-Attribute node has no bearing on the similarity of the Pulley.Weight-Pulley.Weight Attribute-Attribute node; i.e., they are inconsistent and therefore hinder each other (represented by dotted line with two-sided arrows).

Process for calculation of similarity

Processing starts by describing a scene and runs for a certain number of iterations. In the case of Figure 4.3, the scene is made of the two assemblies representing overall input affected objects of two distinct ECs, which are composed of other objects (parts, such as shaft and pulley) and their attributes, such as weight, volume,

manufacturing process name, and so on. For the purpose of explanation, we shall assume that the attributes do not need alignment and are attached to each object directly. Any $A_j < 0.5$ implies that the node j does not have aligned entities, whereas any $A_j > 0.5$ implies that the node has aligned entities. The value of A_j is one of the two:

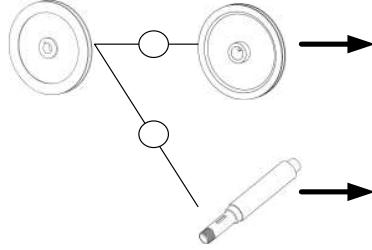
1. Mutually excitatory and inhibitory activations among the nodes. The subscript i identifies the node, and t indicates the time/iteration at which the value is calculated.
2. Match value as a result of finding similarity among the values taken by corresponding attributes. It influences the attribute-to-attribute nodes.

At the start, all nodes are assigned an activation value of 0.5, i.e., maximum uncertainty on the alignment between two entities.

Activation process for object matching: The activation value of a node i at a iteration step $t + 1$ is given as:

$$A_{i,t+1} = \frac{\sum_{j=1}^n R_{ji} W_{ji} S_j}{\sum_{j=1}^n W_{ji} S_j} \quad (4.2)$$

where, R_{ji} in node j is the recommended activation value for node i , and W_{ji} is the weight of this recommendation, and n is the number of incoming nodes. R_{ji} is determined by the type of recommendations-excitatory and inhibitory-and is given as follows: For excitatory connections,



Object-Object node	Attribute node	Attribute i	Attribute j	Mach values
Pulley-Pulley node	Component Name	"pulley"	"pulley"	1
	Weight (lb)	5.45	1.57	0.06
	Volume (cu in)	19.16	5.455	0.32
	Function	Transfer	Transfer	1

	Material name	"AISI 316 SS"	"AISI 316 SS"	1
	Mgf process	Sand casting	Sand casting	1
	Disposal	"Landfill"	"Landfill"	1
Pulley-Shaft node	Component Name	"pulley"	"Shaft"	0
	Weight (lb)	5.45	1.9	0.07
	Volume (cu in)	19.16	6.65	0.35
	Function	Transfer	Transfer	1

	Material name	"AISI 316 SS"	"AISI 1020"	0
	Mgf process	Sand casting	Milling	0.67
	Disposal	"Landfill"	"Landfill"	1

Figure 4.4: Calculating attribute match value as a part of activation process. Pulley-Pulley object node and Pulley-Shaft object node are excited by attribute node, and then the attribute node is excited by match values.

$$R_{ji} = \begin{cases} A_i + (1 - A_i)(A_j - 0.5) & , if A_j > 0.5 \\ A_i - A_i(0.5 - A_j) & , if A_j < 0.5 \end{cases} \quad (4.3)$$

For inhibitory connections,

$$R_{ji} = \begin{cases} A_i + (1 - A_i)(0.5 - A_j) & , if A_j < 0.5 \\ A_i - A_i(A_j - 0.5) & , if A_j > 0.5 \end{cases} \quad (4.4)$$

As mentioned earlier, the values of A are initially assigned to be 0.5. However, the values of the attribute-attribute nodes obtained by calculating the match values will modify the value of A in the next iteration. The process will continue for the desired number of iterations.

Matching function of distinct attributes: The match value for attribute-attribute nodes is calculated based on the type of the attribute as shown in Figure 4.4.

For categorical attributes (a_1 and a_2), such as comparing the joint types of assemblies,

$$matchValue(a_1, a_2) = \begin{cases} 1, & \text{if } a_1 = a_2 \\ 0, & \text{otherwise} \end{cases} \quad (4.5)$$

For numerical attributes, the match value is obtained by using the distance metric. For example, the match value between the weights (w_1 and w_2) of parts is given by

$$matchValue(w_1, w_2) = e^{-\alpha \times distance(w_1, w_2)} \quad (4.6)$$

where $distance(w_1, w_2)$ is the Euclidean distance between the values w_1 and w_2 and the α is decided based on normalization with the maximum and minimum distance for all possible pairs of attributes.

For semantic attributes using domain knowledge, the match value is formulated by assessing ontology of attributes. There are several methods to calculate similarity between two concepts using domain knowledge, e.g., edge-based approach [35, 36], information based approach [37, 38] and contextual information based approach [37]. With a well-developed database, a precise approach can be used to calculate semantic matching value. In this research, a normalized edge-counting approach is used, using the taxonomy of individual semantic attributes. The range of normalized edge-counting is $[0, \infty]$, and it is formulated between the semantic attributes (s_1 and s_2) as:

$$matchValue(s_1, s_2) = \frac{(2 \times MAX - len(s_1, s_2))}{2 \times MAX} \quad (4.7)$$

, where MAX refers to the maximum depth of taxonomy and $len(s_1, s_2)$ is the shortest length between the two concepts of s_1 and s_2 .

For example, Function attributes are defined with FSR taxonomy as shown in Figure 3.4. With the knowledge of taxonomy, the match value of *Branch* and *Channel* function is calculated as a value of 0.67 with the depth of 3 and the length of 2.

EC similarity measure: Once the node activations have been adjusted (identified by a fixed set of iterations or incremental increase in activation value), the measure works successfully in aligned matching between similar products. EC similarity is derived based on MIP (Match In Place) and MOP (Match Out of Place), which are automatically identified during the activation process based on one-to-one matching and parallel connectivity. It enables aligned matching between different cardinality of objects; the overall similarity is given by

$$Sim(A, B) = \frac{\sum_{i \in MIP} (matchValue_i \times A_i) + \sum_{j \in MIP^-} (matchValue_j \times A_j)}{\sum_{i \in MIP \cup MIP^-} A_i} \quad (4.8)$$

, where MIP is the pair of aligned parts and MIP^- is the pair of parts where one of two parts are misaligned, at least in the assignment. The maximum value of 1 is for all product composition the same between two assemblies. The equation considers the similarity of each matched part as well as the difference of connected parts by the term MIP^- . The quantity and quality of misaligned parts influence the overall similarity value.

The similarity values obtained for the input affected objects and output affected objects are then used in equation 4.1 to calculate the overall similarities.

4.5 Parallel engineering change clustering

In this section, the Parallel Engineering Change Clustering (PECC) method has been proposed to identify the distinct subgroup of ECs based on EC similarity explained in Section 4.4. As mentioned in Section 4.2, Parallel Engineering Change Clustering (PECC) is specified for EC similarity measure and the characteristics of Engineering Change (EC).

4.5.1 Clustering method for engineering changes

Clustering method is an analytic tool to group objects or data into the subset called ‘a cluster’ [39]. It partitions unlabeled data with respect to the degree of similarity between objects. The clustering methodologies are varied with respect to data representation, the characteristics of similarity measure, and grouping techniques.

Data representation: There are two common forms of data representations of vectorial representation and pairwise proximity representation [40]. The vectorial representation utilizes multidimensional vector space. Pairwise proximity representation use a symmetric matrix of pairwise similarity measure. EC representation mentioned in Section 4.3 is structured and complex data. The EC data is not available in representing feature vectors in multidimensional vector space. Therefore, the pairwise proximity representation is suitable for the EC clustering method instead of providing real data representation.

Similarity measure: The EC similarity measure described in Section 4.4 is derived from alignment matching, which may not rely on triangular inequality. It precludes the use of traditional clustering methods based on distance-based approach and centroid-based approach. For the alternative, there are some approaches to handle nonmetric similarity measures like median distance approach and embedding strategy [40, 41].

Grouping techniques: Grouping techniques corresponding to EC representation and EC similarity must be carried out with specific environmental condition of EC. First, the number of clusters within EC data cannot be available due to the complexity generated at different sources from design, process, designer, and their relation. Second, EC data is dynamic and evolving over time. A clustering method for ECs needs the capacity to update clusters avoiding high operating cost. Finally, it must effectively work on a large database.

In the next section, Parallel Engineering Change Clustering (PECC) method is proposed to encompass the above requirements for ECs.

4.5.2 Parallel engineering change clustering algorithm

This section provides the detailed description of the proposed clustering method to handle engineering changes. The PECC method is designed based on the ant-based clustering method. The generic ant algorithms were worked by some re-

searchers [42, 43, 44]. Herein, we introduce some modifications to overcome EC environmental requirements. The algorithm we are proposing has resulted in different behavior patterns and improved computing speed from previous ant-based clustering methods.

Basics of ant based clustering

Ant-based clustering is inspired by the emergent property of ant colonies. Imitating the movement of ants, which involves moving food, corpse and larva, individual ants pick up data and transport them to similar data dummy. The algorithm of ant-based clustering were explained in [45, 46, 42]. First, artificial ants are randomly spread out on 2-dimensional data space. Each artificial ant independently explores data space where data are initially scattered at random. They pick up one datum using pickup probability and drop out the datum at a similar data pile. This movement constructs emerging clusters. Behaviors of ants are determined by local density measures of similar objects. The density measure of similar object is represented by probability conversion functions introduced by Deneubourg et al [45].

$$P_{pickup}(i) = \left(\frac{k^+}{k^+ + f(i)}\right)^2 \quad (4.9)$$

$$P_{drop}(i) = \left(\frac{f(i)}{k^- + f(i)}\right)^2 \quad (4.10)$$

, where $f(i)$ refers to similarity density function, and k^+ and k^- are constant parameters for behavior decision. Lumer and Handl et al empirically suggested $k^+ = 0.1$ and $k^- = 0.3$. To demonstrate the robust performance, ant based clustering method

was enhanced by adopting short memory, increasing radius of perception, and spatial separation technique [42].

Neighbor counting

EC neighbors refers to EC data that are regarded to influence similar impacts. It is defined if the similarity value of two EC data exceed a certain threshold value ranged between 0 to 1. The threshold value is a user defined value that can be calculated with the correlation analysis between similarity value and carbon footprint as shown in Section 4.6.1. With a given threshold value of θ , two EC data of EC_1 and EC_2 are defined neighbors by the following:

$$sim(EC_1, EC_2) > \theta$$

The performance of EC clustering aims to gather the neighborhood properly into a cluster. Previous similarity measure of ant-based clustering is based on distance based dissimilarity measure, which cannot capture the EC neighbors properly. In order to pick up low-density neighbors, we propose a new similarity density measure to be adapted to EC neighborhoods. It captures the number of neighbors corresponding to individual EC data within an ant boundary as follow:

$$f(EC_i) = \frac{1}{\sigma} \frac{\sum_{j=1}^n 1_{\Theta}(sim(EC_i, EC_j))}{n} \quad (4.11)$$

$$1_{\Theta}(x) = \begin{cases} 1, & \text{if } x \geq \Theta \\ 0, & \text{otherwise} \end{cases} \quad (4.12)$$

, where EC_i is an EC held by an artificial ant and n refers to the total number of EC neighbors; σ is the size of cells for EC neighbors. The modified measure estimates

the homogeneity and density of mutual neighbors within an ant boundary.

Parallelization in EC clustering

Parallel programming divides a large computation with multi-processing nodes. Multiple processors solve the problem faster than single computing processors by dividing the sequential computing and by running multi operation in parallel.

Parallel Engineering Change Clustering (PECC) makes ant-based clustering parallelized in order to improve computational efficiency on a large database. As shown in Figure 4.5, task parallelization scheme in PECC enables concurrent operation of ants between multi-processors. In this parallelization, each processor independently performs a clustering algorithm with sub-set of ants. The access of shared data enables independent work of individual ants with no communication.

The behavior of ants in PECC may have mutual intervention when two ants concurrently access a set of data in their sequence. For example, in a case when an ant calculates pick-up probability measure, another ant can interfere with the behavior decision. To avoid the race-condition, we propose stygmery to inform the coordinates held by ants in 2D data space, which precludes data access by other ants. This can be realized by adding an indicator to the coordinates. This process is controlled by mutual exclusive barriers to prevent concurrent reading and writing on the same data.

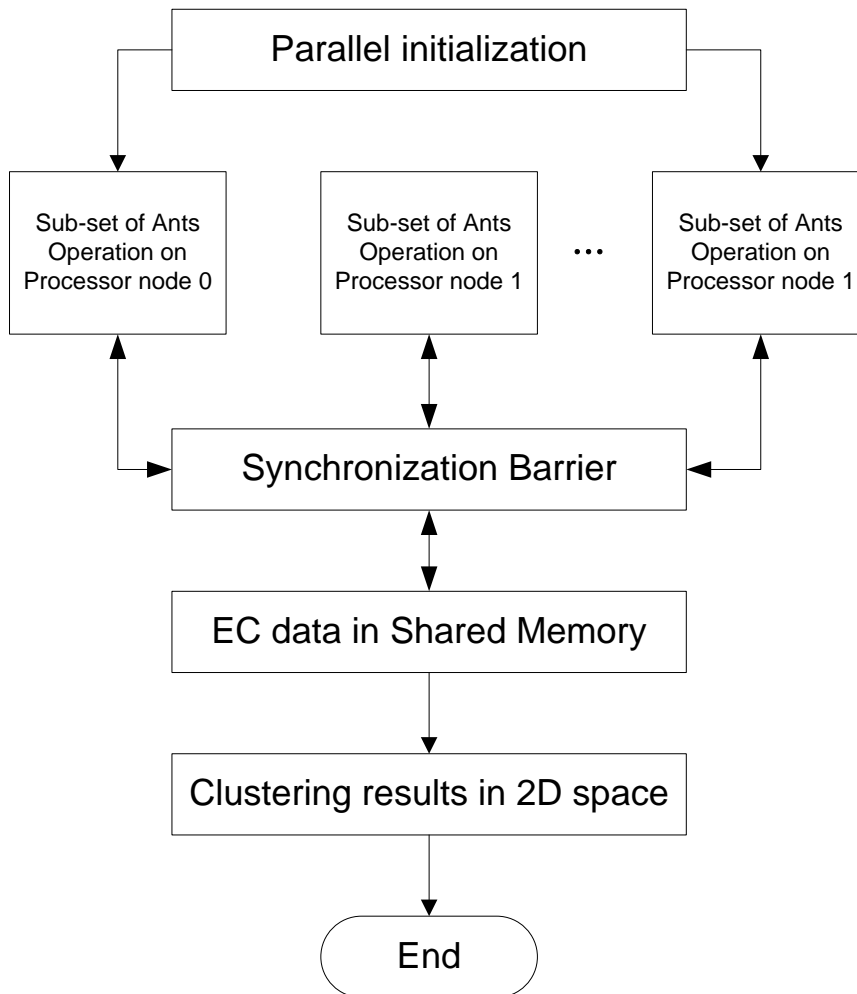


Figure 4.5: Schematic of parallelization in EC clustering

4.6 Evaluation

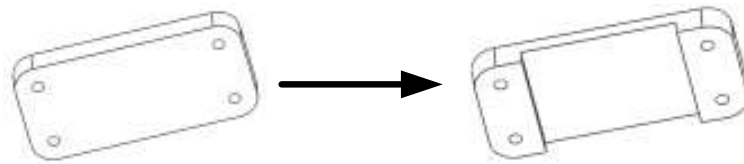
This section presents the evaluation of our similarity measures and the overall approach for clustering Engineering Changes.

The evaluation was conducted using a database of 14 ECs as shown in Appendix A. This results in a combination of 91 matching pairs of problems. The information used from the Engineering Changes is denoted in Table 1. 28 attributes associated with an assembly/part are used. These are spread out across different phases of the product lifecycle, and are selected based on a study of literature and commercial tools. For example, as discussed in [47] (briefly summarized in Figure 4.1), the criteria for sustainability are connected to various impacts on sustainability. Therefore, the criteria that affect sustainability must be defined and formally represented.

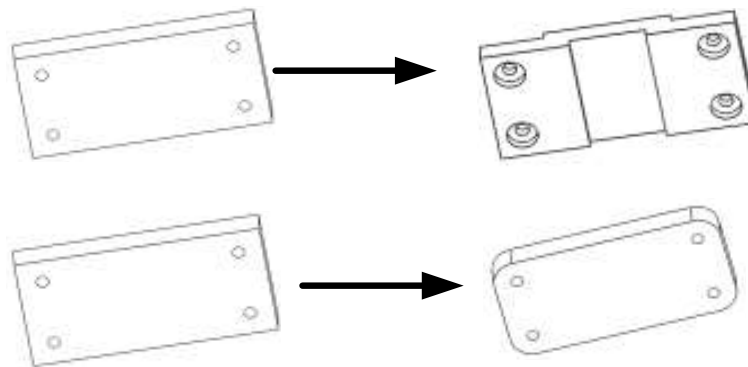
4.6.1 Evaluation of similarity computation

We applied the proposed similarity measure to evaluate quality of retrieval within our data sets. For the sake of evaluation, we have assumed that values of all the attributes are always available. The values of the non-geometric parameters are generated from Cambridge Engineering Selector and Sustainability Xpress tool from Solidworks to ensure engineering relevance.

The resulting similarity values are correlated with the incremental carbon footprint resulting from the change. The carbon footprint values for the input affected objects and output affected objects is obtained using the Solidworks Xpress Sus-



(a) Proposed engineering change



(b) Top two retrieved engineering change

Figure 4.6: A proposed EC and top 2 similar ECs retrieved from database

tainability tool. The difference between the two values gives the incremental carbon footprint resulting from the EC. The evaluation is conducted from two perspectives:

1. A single retrieval case, i.e., one EC matching with 13 others from the database
2. Evaluation of the 91 matching pairs generated from the 14 EC cases
3. Evaluation of the correlation analysis of similarity measure to carbon footprint

Case 1: Finding similarity for one proposed EC. For the proposed EC shown in Figure 4.6 (a), the top two retrieved ECs are shown in Figure 4.6 (b) Figure 4.7 shows the correlation between the similarity values and the incremental carbon footprint. It was observed that the top two retrieved changes (shown in Figure 4.6(b)) have very small incremental carbon footprint, i.e., higher EC similarity implies a higher chance of similar impact on the environment. Some ECs that are dissimilar also have lower incremental carbon footprint, which is fine, since there cannot be claims that dissimilar ECs have very different carbon footprints.

Case 2: Evaluating similarities across 91 pairs of ECs. 91 pairs of matching problems are generated from the 14 EC cases in the database. Figure 4.8 shows the correlation between the similarity value and incremental carbon footprint for only those cases where the similarity was above 0.8. It can be seen the higher the similarity value, the lower the difference in the incremental carbon footprint.

In addition, precision and recall, R-precision, and average precision have been employed to show the overall performance [48]. They are measured for the similar ECs by manual observation.

Table 4.2: Precision and recall (in percentage) for the 91 matching problems in the database shown in the appendix. Threshold value is one above which the changes were considered similar

	Threshold = 0.95	Threshold = 0.90
Precision	100	85.7
Recall	85.7	85.7
R-precision	85.7	
Average precision	87.6	

Table 4.2 shows that we get high precision and recall values for the database. In addition, the higher values of R-precision and average precision imply that most similar ECs can be retrieved in the higher rank; it can be compared to the case that the best results in a web search are obtained on the first page.

Case 3: Evaluation of the correlation analysis of similarity measure to carbon footprint. We have used Pearson product-moment correlation coefficient to analyze the relation of EC similarity to the difference in carbon footprints between ECs. It is defined as the covariance of the two variables divided by the product of their standard deviations, and is one of the most commonly used metrics to evaluate correlations. We get a value of -0.6, which implies that as the similarity value increases, the difference in carbon footprints between ECs reduces. A value between -0.5 and -1.0 is usually considered large in the social sciences. However, we do not have any data in this domain to compare our results with. An additional reason that has prevented a larger negative value of the coefficient is that the correlation is not symmetric, i.e., higher similarity means closer values of carbon footprint, but closer values of carbon footprint does not mean higher similarity between ECs.

The standard t-test was utilized to estimate the statistical significance, i.e., to

Table 4.3: T-test between the difference of carbon footprint between ECs and pairs of ECs that have an EC similarity value over 0.90

$t_0 value$	1.948
$t_{1-\alpha/2}(n-2)$	1.725
$ t_0 \geq t_{1-\alpha/2}(n-2)$	Confidence of 90%

study that this correlation is not by chance. The test statistic is $t_0 = \frac{r}{\sqrt{(1-r)^2/(n-2)}}$, which measures t-distribution with n-2 degree of freedom under the null hypothesis.

Table 4.3 shows the result of correlation analysis. When we consider the relation between EC similarity and CO_2 , EC similarity can be considered to be related to carbon footprint with 90% confidence level.

4.6.2 Evaluation of overall clustering approach

There are no methods for EC clustering method in the related research which could be used to evaluate our approach effectively. Therefore, in this section, Parallel Engineering Change Clustering (PECC) has been evaluated in terms of clustering performance by comparative study and parallel performance through speedup and parallel efficiency. Parameter setting of PECC in this case study follows the recommendation by Handl [42].

To better perform comparative study, clustering methods are considered using the pair wise similarity measure to generate clusters like K-medoids and spectral clustering [49, 50]. Each clustering method employs similarity matrix calculated by EC similarity measure, which data is composed of 42 ECs. K-medoid clustering method is similar to k-means clustering method, but it computes the center of clusters with an observation of data that maximizes similarity measure in a cluster. Spectral

clustering utilize standard linear algebra methods and is based on graph Laplacian of similarity graphs, where ϵ -neighborhood graph has been used.

The comparative study is evaluated with *F-measure* that uses the precision and recall from information retrieval. It will be performed in terms of n number of EC data that is marked by manual observation of each class i . Each cluster j is represented EC_j ; EC_{ij} refers to EC data marked class i placed in cluster j . The overall F-measure for all clusters is represented as,

$$F = \sum_i \frac{n_i}{n} \max_j (F(i, j)) \quad (4.13)$$

The F-measure for a cluster is defined as follows:

$$F(i, j) = \frac{2 \times p(i, j) \times r(i, j)}{p(i, j) + r(i, j)} \quad (4.14)$$

, where precision and recall is calculated as $p(i, j) = \frac{n_{ij}}{n_j}$ and $r(i, j) = \frac{n_{ij}}{n_i}$.

In order to analyze the efficiency of parallel computation, the speedup of parallel computing in terms of 1,2,4 multi-core computing has been evaluated as shown in Figure 4.9. It is calculated by the ratio of running time on one processor and running time on p processors; the speedup is given by

$$Speedup = \frac{t_1}{t_p} \quad (4.15)$$

With speedup, the parallel efficiency is evaluated, meaning the parallelization is implemented considering load balance in computation. It is calculated by

$$Parallel\ efficiency = \frac{Speedup}{p} \quad (4.16)$$

Table 4.4 shows the result of F-measure on each clustering method. The number of clusters are given in terms of the analysis of manual observation.

	K-mediod clustering	Spectral clustering	Parallel EC clustering
F-measure	0.79	0.86	0.97

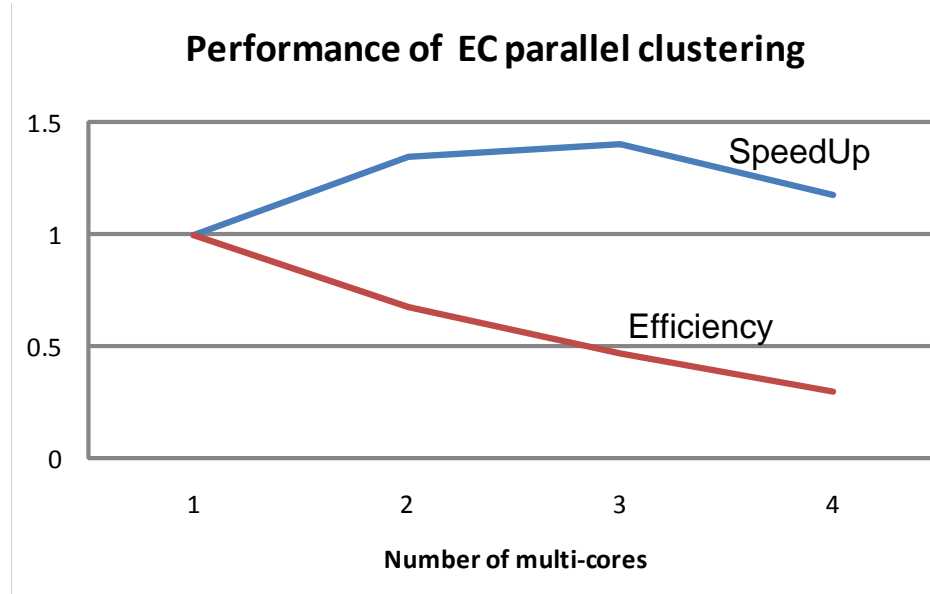


Figure 4.9: Efficiency of core-usage and speed of parallelized EC clustering. Better speeds are obtained with more cores, but the core-usage efficiency decreases; one reason is that the dataset that we used is not large enough

In addition, Figure 4.9 shows that the speedup of PECC is scalable to the number of processors; the load balance is reassigned with an increasing number of ants. The drop in efficiency is severe, because the number of datapoints used in our experiments is not large enough. Future study is needed for searching the parameter for proper parallel nodes.

4.7 Summary

Systematic sustainability assessment of a proposed Engineering Change (EC) is, typically, a time-consuming process due to the complexity of typical products and

the lifecycle-wide impact of a change. In this section, we have proposed a clustering method to enable faster evaluation by providing similar past ECs. From analyzing the representation of engineering changes, EC similarity measure has been defined based on affected object matching. EC similarity utilized an alignment matching approach derived from research in psychology that evaluates the structured data through components and their attribute match values on product information. In addition, EC clustering methodology was introduced to gather ECs that embed similar carbon emission, employing ant based clustering method and parallel computing. Parallelization of clustering methods in the proposed approach offered possibilities for enhanced scalability and throughput on a large database. For example, we applied the measure to a case of 14 Engineering Changes (91 matching problems) and compared the matches for relevance to evaluation of carbon footprint. The precision and recall are evaluated by comparing against carbon footprints obtained using commercial LCA tool. The results justify the initial assumption that similar ECs will have similar impact on the carbon footprint. For parallel efficiency, parallel EC clustering method shows good scalability and load balance.

CHAPTER V

Predicting carbon footprint of proposed new products and engineering changes

5.1 Motivation

Growing concern about global warming and movement for regulation have forced industry to mitigate carbon footprint emission of their products. While companies in previous product development considered optimizing cost and quality with developing time, today's companies need to understand the environmental issues and balance improving carbon impact without aggravating other constraints. The first step toward environmental responsibility is measuring the carbon emission of products and comparing it to the established target.

From the conventional study of environmentally conscious design we can find several approaches to evaluating environmental impacts during the lifecycle. It is known that full LCA study is a successful tool for environmental consideration during product development. In addition, publicly available specification (PAS) 2050 by the British Standard Institute (BSI) provides a quantifying procedure for product carbon emissions. In particular, different simplified methods have been proposed to

integrate with design process. Though these methods provide a quantifying measure of carbon footprint for products, the application to design process is limited to estimating existing products and excludes virtual products in the design process.

5.2 Environmental strategy for developmental process

A company strategy to reduce carbon footprint can be reflected in a new product design and product redesign. For a new product design, designers in the company can achieve the reduced carbon footprint by considering alternative decision decisions like material selection, process substitution, and design layout. Product redesign can contribute to the carbon reduction of environmentally problematic parts through the engineering change process. New products and engineering changes need to be evaluated with incomplete information about virtual products and virtual changes. Ideally, this problem should be solved without delaying design process. To minimize the impacts on developing delay and costs, the environmental estimation should be connected to the process of product development and incorporate the significance of carbon impact as early as possible. However, the LCA method is not linked to design process, and other existing tools are not appropriate in supporting designers' decisions in the design process.

5.3 Carbon footprint estimation for environmentally friendly products

Carbon emission has been the issue of global warming that is being faced as a major challenge. Carbon emission can be easily targeted for mitigation of greenhouse gas emission (GHG) from product lifecycle. In addition, carbon emission is considered to be connected to energy consumption. Therefore, there has been an effort to reduce GHG gas by regulating carbon emission. The footprint refers to the total environmental impacts from product life cycle that involves the scope of raw material extraction, manufacturing process, product use, and disposal. That is, carbon footprint of components in mobile operation provides the carbon emission of components in mobile operation in a comprehensive system boundary of product life cycle.

5.4 Fast prediction of carbon footprint for a new product design and a proposed engineering change

The purpose of this research is to estimate the carbon footprint of a new product and of proposed engineering changes. In the early phase of product design and EC process, predicting accurate environmental impact is not possible because designers cannot provide all related product information due to the degree of freedom in design decision-making; some check lists and guidelines can only be provided. Recent studies, which are integrated with CAD tools, show the possibility of predicting approximate carbon footprint of prototype design after embodiment design process.

However, there are still few methods to support design decision making in the concept design processes and proposed EC analysis. Therefore, our research will focus on the approval of carbon significance in the concept design process and proposed EC analysis process. The decision on carbon significance significantly affects cost and time investments in the later design process. Hence, designers can promote the development process by considering environmentally problematic parts in the early process phase. For fast prediction of carbon footprint for new products and proposed ECs, there are some challenges to be resolved as follows:

1. Incomplete product data in a concept design and a proposed engineering change:
Due to the degree of freedom in design decision-making, it is hard to define the exact impacts of a given concept design and proposed EC. Therefore, we propose the use of knowledge from previous product information and ECs.
2. Lack of emission factor data to represent the relationship between product parameters and environmental impacts: Emission factors refer to the quantity of carbon footprint relative to a unit of activity. For accuracy, the primary data, which is measured by all stakeholders, is required. In addition, secondary data based on generic measurement of similar material or process can be substituted in the absence of primary data. However, a single company cannot afford and maintain all emission factors and apply them into product development due to processing cost and time.
3. No exact same product developments and ECs: No identical product develop-

ments and ECs are processed during the product life time. If the case occurs, information retrieval works well enough to estimate carbon footprint.

5.5 Knowledge-based carbon footprint prediction

To estimate carbon footprint using incomplete activity data, we propose a knowledge-based approach. Knowledge-based approach can facilitate fast carbon footprint prediction without building detailed process maps and using emission factors for unit-processes. The knowledge-based prediction of a new products and proposed ECs has three issues. The first issue is uncertainty of product realization on design parameters. The second issue is developing a method to enable fast prediction of carbon footprint. The third issue is the compatibility in the application of the method to a concept design and to proposed engineering changes.

For the reuse of previous knowledge, we retrieve knowledge of similar cases from a database and adjust concept design and proposed ECs into them. The method is based on case-based reasoning, a problem solving technique to reuse previous knowledge, as shown in Figure 5.1 [51].

The overall framework consists of retrieval and adaptation. As the general process of case-based reasoning, we will retrieve similar products and similar ECs using semantic query and EC clusters with material query. Next, incomplete activity data due to undecided design parameters in the concept design process are estimated by the reuse of similar products and engineering changes. Finally, the carbon footprint of a new product and proposed EC are estimated by adapting the previous carbon information. This research provides the procedure of knowledge based carbon foot-

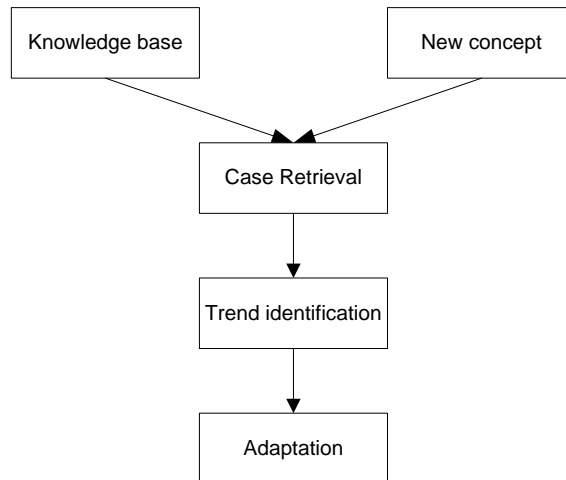


Figure 5.1: The process of case-based reasoning

print for components in mobile operation in the following sections.

5.6 Overall approach

Carbon footprint estimation encompasses from raw material extraction to product disposal. The model estimates key activity parameters as input data with similar cases and decides the significance of carbon footprint as an output. The schematic of knowledge based carbon footprint classification is shown in Figure 5.2. The framework consists of three steps as follows:

1. **To identify key activity parameters:** Activity parameters are used to calculate carbon footprint with emission factors, which represent each phase of the lifecycle. For example, activity parameters are associated with material flow in raw material extraction, energy intensity in manufacturing process, energy consumption in use phase and recycling ratio in the disposal phase. All activity parameters are not readily known in concept design parameters and

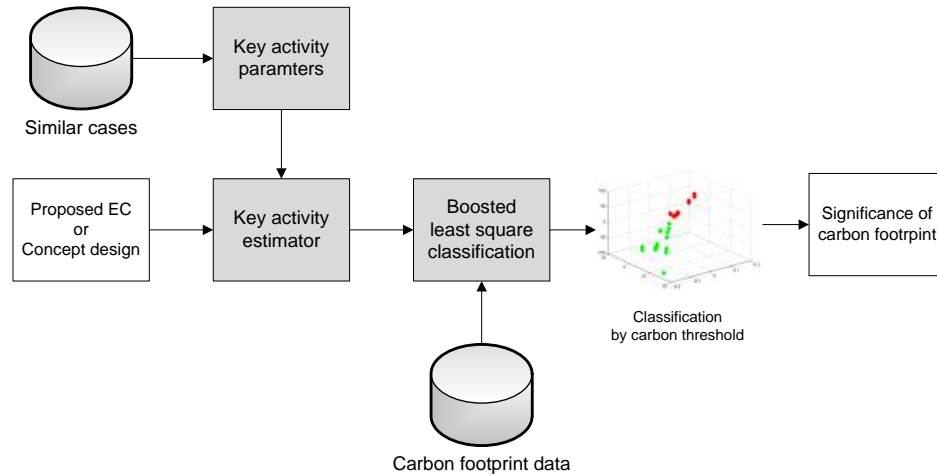


Figure 5.2: Flow chart of knowledge-based carbon footprint classification

proposed engineering changes. In addition, a single company is unable to look into related emission factors due to additional cost and development delay. Therefore, key activity parameters need be proposed to represent the scope of the product life cycle.

2. **To define estimator for activity data using case based reasoning:** Estimators predict the value of key activity using previous product information or engineering changes. Based on case-based reasoning, estimators retrieve similar cases in terms of material and related parts, which is known activity information from concept design and proposed engineering change. Next, key activity parameters are estimated by using the retrieved cases.
3. **To develop carbon footprint classification using previously evaluated carbon footprint and key parameters in a basis of case-based reasoning:** PAS 2050 has explained the process of carbon calculation as a linear combination of activity parameters and emission factors. Though all emission

response is not strictly linear, LCA method and EIO-LCA are also based on linear relations. For simplicity, emission factors for key activity parameters can, therefore, be calculated by using linear regression. Because key activity parameters are abstract and concise, simplified emission factors may lead to the wrong decision. However, boosted simplified emission factors can make better decisions on carbon classification.

5.7 Detailed procedure of knowledge-based carbon footprint classification

The objective of the research is to decide the significance of carbon emission based on known information about concept design and proposed engineering changes. To compare carbon level there must be common functional units and system boundary between carbon footprint data. The system boundary encompasses the lifecycle of vehicle as a complete assembly; this dominates system boundaries of components, subassemblies, processes, materials, and activities involved in the vehicle.

5.7.1 Information from concept design and engineering changes

Concept design and proposed engineering change cannot provide all activity data. For the available information in concept design, general design methodologies have defined the scope of product information as shown in Table 5.1. The concept design phase can provide the functional requirements, simple components with basic layout, and their arbitrary material selection with material mass. The required information for fast prediction on environmental impact, such as geometry, material, manufac-

Table 5.1: Available information in the design process adopted from [1, 2]

Design process	Information available
Needs identification	Functions, Range of specification
Concept design	Functional requirements, Performance, Basic Layout Components and material
Embodiment design	Manufacturing process, Use profile, End of life strategy refinement of design
Detailed design	Decision of all specifications

turing process, and transportation information, is, however, relatively unavailable.

In addition, the proposed ECs involve information about the process knowledge of ECs, problems of changes, affected parts, and change description as shown in Figure 4.1. Among this information, the information of affected parts can inform activity data. However, the proposed ECs do not have all information. The carbon footprint of engineering changes is calculated by the different carbon footprint of input affected parts and output affected parts. The proposed engineering changes don't include product information relating to output affected part information. The information of affected parts in proposed EC provides before-change parts, but after-change parts are unimplemented in the proposed EC. Nevertheless, the proposed ECs have preliminary mass change information about changed parts, detail design of added parts, and change description provided from the EC developer. The change of product feature needs to be considered.

5.7.2 Key activity parameters

Activity data refers to all material and energy flows used to calculate a carbon emission, which can be representative for activities during each lifecycle phase. However, concept design and proposed engineering changes cannot provide all required activity data to calculate a carbon footprint. In this research, we therefore propose the concept of key activity parameters that facilitate estimating the carbon emissions of lifecycle phases over the concept design and proposed engineering change. They can be a material flow, energy consumption, and related physical parameters to each phase of the product lifecycle. For the purpose of representative features, key activity parameters should not only be derived from design attributes of previous product information and engineering changes, but they can also allocate carbon footprint of each lifecycle phase based on physical causality.

Raw material processing: The lifecycle phase refers to the acquisition of natural resources and the process required for material production. Emission factors for the material production are collected based on material information, and are varied depending on material types and production methods, technologies, infrastructure, and geological differences. Primary activity data for material production is the mass of material, which includes information about intermediate material and waste material. Within emission factors in the raw material processing phase, the activity data typically consists of material input, product output, co-product, and waste material. Among the activity data, available information in the product design system is ma-

terial mass flow of products. In addition, the activity data can be allocated based on output material flow. Therefore, we focus on material mass flow of products as the key activity parameter for raw material extraction.

Manufacturing phase: Manufacturing phase consists of the production and assembling process of sub-parts and components. Carbon footprint during manufacturing phase is calculated with the total emission of a set of manufacturing unit processes; this is mostly based on material flow, manufacturing energy consumption, and manufacturing process. The typical emission sources are generated from stationary combustion, mobile combustion, process energy consumption, coolant flow and indirect emission from electricity consumption. Manufacturing energy consumption has a physical causality of part material flow, manufacturing process, and operating parameters; all are used in allocating the carbon basis. Here we are using manufacturing energy for a key parameter. There are also some research to support using manufacturing energy as an allocation basis of carbon footprint in the manufacturing phase [52, 53, 54]. In addition, manufacturing energy can be estimated from product data using process information and CAM software. Therefore, manufacturing energy is used to represent one key activity in the manufacturing phase of this research.

Use phase: Use phase involves energy and waste flow during vehicle lifecycle. The main contribution of use phase in mobile operation is fuel consumption. The fuel efficiency of vehicle is based on fuel efficiency of a vehicle, part mass, and use

time. For the product consuming electricity, use profile through life time is critical in estimating carbon footprint. However, because there is no standard for energy consumption, this research does not consider use energy consumption. We just focus on fuel consumption during the use phase of a vehicle.

End of life phase: The different recycling strategies are applied for different materials and parts. End-of-life strategy includes landfill, incineration, recycle, and reuse. The emission source of this phase contains the energy consumption of dismantling, shredding, separation, and transportation. However, detailed information about disposal strategy is unknown in the product design system. In this research, the end of life information is provided by binary data about recycle or landfill.

5.7.3 Estimator for key activity data of proposed engineering changes and concept design

Estimator for key activity data of proposed engineering changes and concept design is stated as:

For: Products clusters of functionally similar products

or engineering change clusters

Given: Concept design information of a new product design

and proposed engineering change

Estimate: A set of key parameters for concept design and proposed engineering change

The domains of the overall problem incorporate the two different disciplines of concept design and engineering changes. However, they have a common basis in that their embodiment is not implemented. In addition, they have to be analyzed to delineate the product realization when considering sustainable product development. In this research, we provide a method to support the two problems with the same procedure.

Preprocessing to select a cluster for similar products with the same functions or similar engineering changes. Preprocess gathers similar information to estimate the input information as a retrieval stage in case-based reasoning. The reason why we use similar cases to estimate information is lack of information in concept design engineering changes. Similar cases involve domain specific knowledge used before. In addition, designers mostly refer to them to initialize product development.

For engineering changes, we have already defined EC similarity measure and EC

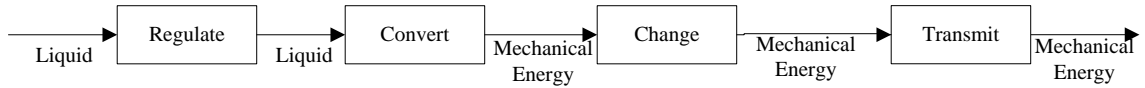


Figure 5.3: Functional model for brake system

clusters with the parallel ant-based clustering method. The EC information in EC clusters are used to adapt this key parameter of proposed engineering changes. For concept design, we have designed functional semantic retrieval to find functionally similar products. However, functionally similar products do not always have similar product embodiments because there may be different realization of functional requirements. For example, the brake system of a vehicle can be initialized from functional modeling as described in Figure 5.3:

With a functional model, similar products can be retrieved by semantic retrieval in FSR (functional semantic representation). In order to classify similar products among functionally similar products by concept design, the amount of flow value using *hasMagnitude* property has been used. Functionally similar products with similar amount of flow may have different material composition. For this, a material matrix has been used. The heterogeneity of material composition is summarized in the material matrix in which 1 is placed if the product has the material and 0 is placed otherwise. Note that a material matrix is created with concept design C , functionally similar product p_n , and material information Mat_m as shown in Figure 5.4:

	Mat_1	Mat_2	...	Mat_m
C	1	1	...	0
P_1	1	1	...	0
P_2	0	0	...	1
...
P_n	1	1	...	1

Figure 5.4: Material matrix

From the material matrix, products are selected to achieve a cluster by calculating Euclidean distance between concept material and product material. The optimistic query is defined with the distance of 0 because the homogeneity of product information leads to more similar product cluster. However, there are tradeoffs between distance value and the amount of product information. In this research, zero distance is selected to maintain homogeneity of material composition with concept design.

5.7.4 Estimating key parameters of concept design and proposed engineering changes

The retrieval uses functional information about concept design. However, estimating carbon footprint requires embodiment of product, which needs mass and material at a minimum. We have to consider the context of product information for embodiment information in retrieval. For the adaptation of the retrieved products, there are some methods like majority vote, 1-NN, weighted sum with KNN, and parallel adaptation with linear regression. The methods are based on the comparison between given information and similar information from a database. The comparison

is not applicable in concept design, because product information in concept design is undefined yet. From the common information between a given concept design and proposed EC, we will show how to consider part by part and phase by phase to derive key parameters. For the estimation of carbon footprint to components in mobile operation, the scope of product lifecycle, product key parameters, assumptions, and data sources are provided as follows:

Material mass: Mass information about concept design and proposed ECs are obtained by the component level. Concept design and proposed ECs are commonly composed of a set of components. However, the material used in components has different energy and material flows in raw material production. Considering the mass impact of material in carbon estimation, the components from a database are classified and used with the same material as components of concept design and proposed engineering changes, as shown in Figure 5.5.

Manufacturing energy: Manufacturing energy is also collected by component level. It refers to the sum of manufacturing process energy to produce a component. Manufacturing energy consumption is required to build a process map with activity data to calculate the carbon footprint. However, there is no process information in concept design and engineering changes. The information available consists of mass information about related components. To predict the manufacturing process of change impacts, we will adapt the manufacturing energy with respect to mass

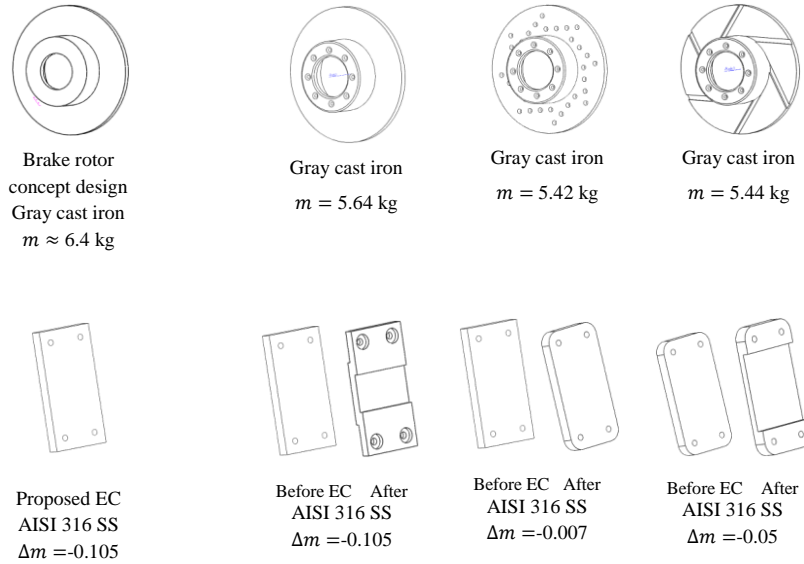


Figure 5.5: Product and engineering changes that have the same material composition

information of concept design and similar ECs. For example, a cluster of brake rotor from our database is analyzed with respect to manufacturing energy change in Figure 5.6.

To predict the manufacturing energy consumption, manufacturing energy consumption is linearly approximated with respect to mass information. Next, the adaptation for manufacturing energy consumption employs parallel adjustment to correspond to linear regression between material vector and manufacturing energy consumption. It adjusts manufacturing energy of concept design by the average of a parallel line from k-neighbors.

Fuel consumption: Fuel consumption is calculated based on mass information and fuel consumption of the target vehicle. Fuel consumption in the use phase is dependent on the performance of products, but it is not just allocated to part

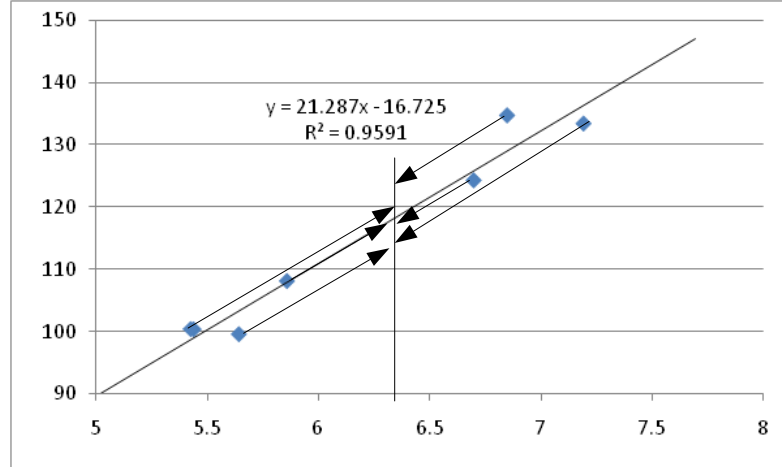


Figure 5.6: Manufacturing energy adaptation for a brake rotor

design or changes. Because the contribution of functional improvement in part design and changes may influence complex results, the impact is hardly predictable. We just consider the impacts of concept design and proposed ECs to affect product performance with respect to their mass information. This process is calculated based on vehicle performance.

During the lifetime, we can estimate fuel saving from the above regression model that is based on 37 gasoline cars. The regression function for fuel consumption (FC) is represented with respect to car curb weight (CW) as shown in (5.1).

$$FC = -10^{-7}CW^2 + 0.0026CW - 1.788 \quad (5.1)$$

The regression model is obtained with 37 reference vehicle models in order to estimate the fuel use of the redesigned car in terms of car weight. We will consider the fuel saving (FS) from the mass change using the regression model:

$$FS = \frac{dFC}{dCW} \Delta mass \quad (5.2)$$

With the performance data of a reference car, the fuel consumption from concept design and proposed engineering change can be obtained.

Recyclability: End of life strategy is a binary variable to select options like recycle and landfill. We will just use the majority vote to decide the disposal strategy of components related to concept design and a proposed EC. The impact of disposal strategy is different with respect to material and parts, which is not directly defined from product information and EC information. Therefore, we will define qualitative information to derive classification.

5.7.5 Carbon footprint classification using previously evaluated carbon footprint and key parameters

The section aims to formulate a classification method to predict the significance of product carbon footprint to recognize the measure for decisional and selective purposes. The decision about carbon footprint is made based on the analysis of previous carbon footprint data of functionally similar products and similar ECs. It integrates material composition, product mass, manufacturing energy consumption, fuel consumption and recycling strategy into the significance of carbon footprint. Though the emission generated from each lifecycle phase cannot be estimated explicitly with key parameters, it is useful in integrating environmental impact with design development without building a complex process map in the early phase of design.

Overall algorithm

The overall problem of carbon footprint classification is stated as follow:

For: Key parameter of given concept design and proposed ECs

Given: Similar cases with their carbon footprint and target carbon emission

Determine: Significance of carbon footprint of concept design and proposed ECs

Carbon footprint is calculated using the equation that is the sum of carbon emission generated from all materials, energy and waste for activities over the product lifecycle. The calculation is a simple sum of multiplying activity data by emission factors. From analogy to carbon footprint calculations, the linear classification model is simply applied from learning with previous carbon data. There have been several approaches using a linear estimation concept in simplified LCA studies. However, the studies cannot provide the prediction of carbon footprint with high level information derived from concept design and proposed engineering changes. In addition, though the carbon footprint calculation is a kind of linear model, it is not a deterministic function because there is mandatory decision making by the stakeholder, by geometrical and temporal differences, and by data sources. A linear regression model for predicting carbon emission is, therefore, a weak estimator. To reduce the uncertainty of the prediction, the combined decision making from different models learned with previous data has been considered in this section.

Adaboosting algorithm with random linear classifiers: AdaBoost was

proposed by Freund and Schapire [55]; it aims to make a strong classification method by using majority vote principle of weak classifiers. The Adaboost algorithm takes an input as an instance of domain data with the label. In case of binary classification, the label is decided as one of the two values: -1,1. With the training data, Adaboosting algorithm decides the distribution of weak base classifiers. The weights of weak classifiers are adapted in order to minimize the risk of a combined decision. Initially, the weight of weak classifiers are equally set. Adaboosting algorithm increases the weight to correct classifiers or decreases the weight to incorrect classifiers. It can specify the weighted majority vote of weak classifiers using previous data.

Linear classifiers as base classifiers provide a decision about the significance of carbon footprint. Random numbers selected from training data are used to estimate linear classification by using least square method. A linear combination of key activities can provide abundant decision boundaries. A linear classification function using key activity parameters is defined by

$$\text{sign}(f(x) - CO_{2_{target}}), \text{ where } f(x) = a_1x_{mass} + a_2x_{mfg} + a_3x_{use} + a_4x_{recycle} \quad (5.3)$$

The parameters of $a_1, a_2, a_3,$ and a_4 are generated from least square method using randomly selected training data.

Impact classification: The carbon footprint of an automotive component has to be targeted by environmental experts or designers. The target value should be established by minimizing the carbon footprint emitted during the product life cycle. With the target carbon footprint value, we can decide the significance of environ-

mental impact on given activities. The simple stump will be defined in terms of all EI attributes, which will be boosted to define ΔCO_2 classifiers as follows:

$$y = CO_2 - CO_{2_{threshold}} \begin{cases} \text{high impact, if } y \geq 0 \\ \text{low impact, if } y < 0 \end{cases} \quad (5.4)$$

We can consider the variation of k to estimate ΔCO_2 of proposed EC. In addition, it can provide the feasible EI range of other design decision-making by experts' knowledge.

The output of estimating lifecycle impact of ECs: Boosting algorithm determines the amount of all metrics to classify carbon footprint with the given threshold. We will define the upper bound of carbon footprint to predict the significance of carbon change. Boosting algorithm is a kind of ensemble classification method, which uses a set of basic classifiers to identify data information. Basic classifiers represent simple classification of data using partial information. In our method, four types of basic classifiers are employed as follows: material classifier, manufacturing energy classifier, use energy consumption classifier, and recycle classifiers. Classifiers will be varied based on their threshold value.

5.8 Evaluation

Performance of boosting method: 0.632 bootstrap estimation with respect to various threshold values: This test focuses on the classification of the carbon footprint of EC into significant impact or insignificant impact. The threshold value

of the classification will be given by the domain experts to manage carbon footprint of ECs. To classify carbon footprint of ECs, weak classifiers on material mass, manufacturing energy, fuel saving, and disposal option are provided. 0.632 bootstrap is proposed by Efron and Tibshirani [56] in order to reduce the bias of the leave-one-out bootstrap. The estimate has the form of

$$\widehat{Err}^{.632} = 0.368 \times \overline{err} + 0.632 \times \widehat{Err}^{(1)} \quad (5.5)$$

The results downward bias when there are no class differences. $W = 0.632$ is proposed by Efron. 0.632 estimator works well with a small number of training data. $\widehat{Err}^{(1)}$ is the error of leave one out bootstrap over n number of training data. The \overline{err} is the average of training error during bootstrap. For the sampling data, at least one observation is sampled from each class.

Performance of boosted least square classification using random samples: The threshold of classification will be decided by a domain expert. Therefore, we will show the bootstrap errors with respect to the value of threshold and the types of classification.

We have considered 0.632 bootstrap of Adaboost with linear classifiers, where the bootstrap sample $B = 200$ with training iteration $T = 500$. For the bootstrap evaluation, 51 samples are used and labeled with threshold value. Figure 5.7 shows the 0.632 bootstrap error rate with our boosting method and other typical classification methods. The boosting method overall outperforms the other classifiers with

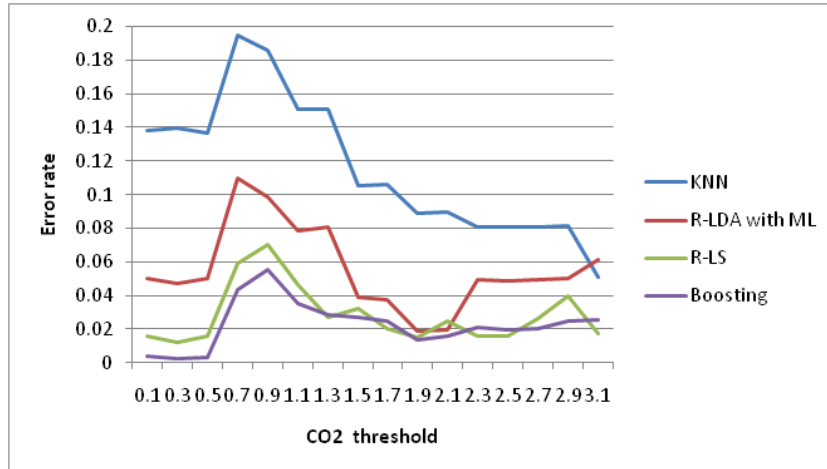


Figure 5.7: 0.632 Bootstrap error rate with $B = 200$ for EC carbon data

respect to the threshold value. In addition, the boosting method showed minimum error rate during the test.

5.8.1 Case Study

In this section, we demonstrate a case study to show the application of knowledge-based carbon footprint estimation for concept design and proposed engineering changes. It consists of two examples for concept design and proposed engineering changes using an automotive brake system as an example for concept design and proposed engineering changes. Data has been created with arbitrary implementation of mechanical elements and their changes. The carbon data has been calculated with SolidworkSustainXpress for material, manufacturing, transportation, and disposal phase and CES2010 ecoaudit for use phase. SolidworkSustainXpress is integrated with CAD system to calculate carbon footprint with material information and geometry information. For the use phase, we have assumed that the machine elements are

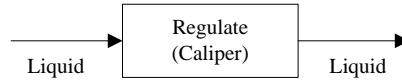
used in mobile products and that the technical improvement and functional change are not dealt with. Because the technical improvement and functional change can be considered to be brought with detailed information to estimate the impact, they are separately considered.

Product system

The brake system of a family car is analyzed to decide the significance of carbon footprint. The functional unit of a vehicle is defined with 10 years of life time and 10,000 miles per year. The system boundary encompasses subcomponents of the brake system over the entire life cycle. In particular, concept design for brake system just considers a subcomponent of the brake system: the caliper part. Engineering change is the change of brake rotor in shape and materials. In this research, a reference lifecycle of vehicle has been defined with a family car, whose functional life span corresponds to 100,000 miles during 10 years.

Preprocessing

To retrieve similar product information for concept design and proposed engineering change, the system utilizes the functional information for concept design as well as input-affected product information for proposed engineering change. Functional modeling of brake caliper using FSR, which can represent the semantics of regulating braking power from hydraulic pressure, enable semantic retrieval to provide functionally similar products as described in Figure 5.8, which functional information was adopted from the Design Repository at Missouri University of Science and Technology [28].



$$\text{Regulate}(\?x) \wedge \text{hasInputFlow}(\?x, \?y1) \wedge \text{Liquid}(\?y1) \wedge \text{hasMagnitude}(\?y1, \?force) \wedge \text{Swrlb:greaterThan}(\?force, 800) \wedge \text{swrlb:lessThan}(\?force, 1000) \wedge \text{hasOutputFlow}(\?x, \?y2) \wedge \text{Liquid}(\?y2) \rightarrow \text{sqwrl:select}(\?x)$$

Figure 5.8: Functional model of brake caliper assembly and semantic retrieval using FSR

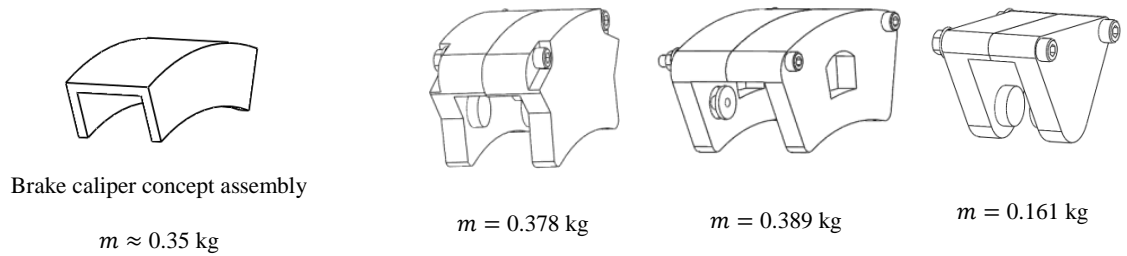


Figure 5.9: Retrieved examples of caliper from semantic retrieval and material matrix

In order to find similar calipers, we have utilized the force of liquid as the amount of *Liquid* flow with *hasMagnitude* property. It enables designers to consider functional similarity as well as similar performance of products. Retrieved products may have different material composition. By using the material matrix, we can select the functionally similar calipers with the same material composition. Calipers are selected from concept design as shown in Figure 5.9.

Key parameter estimating

In this approach, key parameters of concept design and proposed engineering changes are estimated from given product and engineering change clusters. Key parameters consist of material mass, manufacturing energy consumption, fuel consumption in mobile use, and end of life strategy. Different estimators are used to estimate each key parameter as follows:

Mass information: Retrieved similar brake systems similar to the brake concept design and proposed EC of brake rotor provide the material flow information on each component. The information of material mass is summarized with a vector format. Each attribute of the vector is filled with mass of the material given by concept design or proposed EC, if concept design provides the material information.

Manufacturing energy consumption: The estimator for manufacturing energy consumption employs parallel adjustment to correspond to linear regression between material vector and manufacturing energy consumption. For example, rotor and caliper of concept design provide material flow. The material information determines manufacturing energy consumption of each component that is adjusted parallel to linear regression from the nearest neighbor. The proposed engineering changes are also processed by parallel adjustment. The result is shown in Figure 5.10.

Fuel consumption in use phase: with the functional unit of the vehicle, the fuel consumption from concept design and proposed engineering change are calculated using the equation 5.1. The regression model is obtained with 37 reference

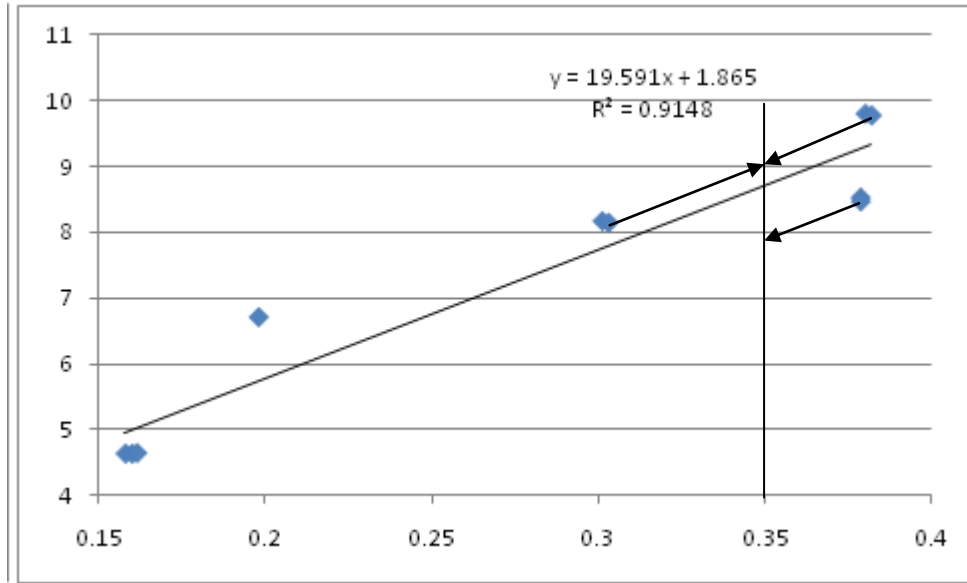


Figure 5.10: Adapting process of manufacturing energy consumption

vehicle models in order to estimate the fuel use of the redesigned car in terms of car weight. We will consider the fuel saving (FS) from the mass change using the regression model with the equation 5.2.

End of life strategy: The end of life strategy is different with respect to material and parts, which are decided by end of life strategy of similar calipers. The decision making is defined from the majority vote from EOL strategy of similar products. The retrieved caliper assemblies are considered to be recycled in the end of use.

$$EOL = 1$$

Boosted learning surrogate LCA method

The carbon should be targeted to each component and provided to train the boosting algorithm. The carbon target encompasses carbon emission over the product lifecycle. In this case study, the targets were established with the average carbon footprint of previous similar products. The significance of carbon footprint during

Table 5.2: The prediction of carbon significance on new products and proposed ECs

	Rotor design	Caliper design	EC of base part
Mass (kg)	6.4	0.35	-0.1
Manufacturing energy(MJ)	119.51	8.73	8.308
Fuel consumption(lb)	296.3	16.2	-4.629
EOL	1	1	1
Target CO_{2eq} (kg)	38	5.5	-2.5
Significance	Insignificance	Insignificance	Insignificance

lifecycle is calculated with material mass flow, manufacturing energy consumption, fuel consumption, and EOL strategy as shown in Table 5.2.

The carbon footprint of components in concept design is considered a sum of carbon emissions along the sub-parts over the life cycle. Each carbon classification with boosting margin provides the carbon significance with the confidence of correctness, which enables designers to identify problematic parts and benign parts. The result provides the carbon footprint of product realization from concept design.

5.9 Summary

As the issue of carbon footprint has been a challenge to modern companies, the contribution of designers has been critical in developing low carbon products. This trend promotes the need of environmental tools for designers and engineers. To date, some methodologies have been developed to support environmental consideration in product development. However, the existing tools are too complex or too qualitative to meet the environmental needs of designers. In particular, there is lack of analytic approach for the early design process, where 70% of product design is determined.

This research proposes a framework to predict carbon footprint of products in the early design process. As a challenge of this approach, lack of knowledge available in the early design process was identified, and the needs of knowledge-based approach was shown. This proposed method introduced a way to reuse carbon footprint information deriving from similar existing products and engineering changes. This was implemented by using case-based reasoning and boosted learning surrogate LCA. As an input, key parameters were determined to represent lifecycle attributes. This enables designers to estimate the feasibility of target emission to virtual design. The result of our approach has been shown to perform well with carbon footprint data. It is believed that the key parameters can be correlated with carbon footprint emission. However, the key parameter was determined from literature review and intuition. Future work is needed for selecting suitable key parameters and verification method with testing.

CHAPTER VI

Conclusion

This chapter summarizes key contributions of this research and discusses directions for future research.

6.1 Research contributions

Current approaches to assess sustainability (carbon footprint in this thesis) rely on detailed computations possible only after complete knowledge about the product or engineering change is generated, and are, therefore, time-consuming, off-line and reactive (post-design or post-engineering change). As a result, they are not integrated into the product creation process prior to finalizing details and committing resources for downstream activities. This dissertation presents research focusing on the development of knowledge-based techniques to transform the current approach into an integrated proactive approach that relies on using fast estimates of sustainability generated from past computations on similar products. The developed methods address multiple challenges by leveraging the latest advancements in open standards and software capabilities from machine learning and data mining to support integration and early decision-making using generic knowledge of the product development

field.

We can determine the environmental impact of a new product using functionally similar products, because Life Cycle Assessment (LCA)-based carbon footprint calculation typically starts by analyzing the product functions. However, the lack of a semantically correct formal representation of product functions is a barrier to their effective capture and reuse. Existing representations are unsuitable for automated reasoning tasks, since they capture only a restrictive, practically unusable set of semantics determined by the limited expressive power of the Web Ontology Language (OWL). The Function Semantics Representation (FSR) proposed in this research is developed by first identifying the advanced semantics that must be captured to ensure appropriate usability. These are then represented using Semantic Web Rule Language (SWRL), a proposed Semantic Web standard. As demonstrated through multiple cases, this ensures support for an effective reasoning mechanism to develop and validate the product function (or functional model). Furthermore, it enables finding similar products to predict/estimate the carbon footprint of a proposed product design.

Several products are developed as engineering changes of previous products but enough data to predict the carbon footprint is unavailable before their implementation. Using past EC knowledge to predict the carbon footprint of a proposed EC requires an approach to compute similarity between ECs. We proposed an approach to compute similarity between ECs that overcame the challenge of the hierarchical nature of product knowledge by integrating an approach inspired from research in

psychology with semantics specific to product development. We embedded this into a parallelized Ant-Colony based clustering algorithm for faster retrieval of a group of similar ECs. We applied the approach to 14 Engineering Changes (91 matching problems) and compared the results from the perspective of carbon footprint evaluation. We show that there is a statistically significant improvement in precision in retrieving similar ECs using our approach as compared to those using state-of-the-art approaches.

We are not aware of approaches to predict the carbon footprint of an EC or a proposed design right after the proposal. In order to reuse carbon footprint information from past designs and engineering changes, key parameters were determined to represent lifecycle attributes. The carbon footprint is predicted through a surrogate LCA technique developed using case-based reasoning and boosted-learning. We evaluated against state-of-the-art approaches to observe that there is a significant improvement in success rate in predicting carbon footprint obtained using our approach as compared to that obtained using the state-of-the-art approaches.

6.2 Future research directions

Following are some of the interesting directions that this research can be extended to:

- **Integrate the three dimensions of sustainability:** We considered techniques to integrate carbon footprint evaluation into the early phases of product creation. Sustainable manufacturing aims to achieve a balance along three, sometimes conflicting, dimensions: environmental, economic, and societal. This

research needs to be extended to develop appropriate knowledge representations, metrics, and integration techniques to ensure that the three views are presented to the designer in a real-time, seamless manner to enable monitoring and assessments for sustainable manufacturing.

- **Determining an optimal number of datasets for prediction:** The prediction of carbon footprint, as discussed in this research, relies on past knowledge. Whereas creating large number of datasets is a prohibitive task, using a small number of datasets could reduce the reliability of the prediction. Therefore, research is required to determine an optimal number of datasets required for reliable prediction.
- **Presenting similarity in a human-understandable form:** In the product development domain, the mathematical values and the geometric definitions of similarity are not intuitive, because formal mathematical models have not been developed for capturing human perceptions. Domain experts cannot easily assign a real value to their perception of similarity. Research is therefore necessary to determine a format for presenting the results such that the domain experts find them more usable and acceptable. Candidate approaches include presenting a ranked order of the results and classifying into categories, such as “very similar”, “similar”, “dissimilar”, “very dissimilar” and so on.
- **Managing scalability:** Methods proposed in this thesis are designed to use parallel computing power presented by modern computers. Further research is required to analyze and modify the proposed methods to handle scalability,

particularly at the scale of automotive product knowledge that is very large and complex (millions of attributes per dataset and millions of datasets).

- **Development of a repository:** We have used our expert knowledge to determine the correctness of the results for the demonstrative cases that we have developed. Standard data sets should be generated from complex, real-world products and should incorporate multiple views of representation and the contribution of manufacturing processes, materials, and other parameters affecting carbon footprint. Such rigorously developed datasets can form a cyber-repository that can be used to benchmark/validate different methods and tools developed by researchers in this field.
- **Enabling decision-making at a finer level of granularity:** In this research, environmental impact was assessed using the standard, single-valued carbon footprint. It is desirable to develop methods to integrate predictions at a finer level of granularity for effective decision-making. For example, the designer might be interested in those phases of the product lifecycle that would bear the most negative/positive impact. Similarly, the designer might want to identify those components, e.g., material, manufacturing processes, etc., that contribute the most to the overall impact.

Bibliography

- [1] M. Kutz, *Environmentally conscious mechanical design*. Hoboken, N.J.: John Wiley, 2007.
- [2] C. Telenko, C. C. Seepersad, and M. E. Webber, “A method for developing design for environment guideline for future product design,” in *IDETC/CIE, San Diego, California, US*, 2009, p. 12.
- [3] K. Capoor and P. Ambrosi, “State and trends of carbon market,” The World Bank, Tech. Rep., May 2009.
- [4] M. Erden, H. Komoto, T. van Beek, V. D’Amelio, E. Echavarria, and T. Tomiyama, “A review of function modeling: Approaches and applications,” *AI EDAM*, vol. 22, no. 02, pp. 147–169, 2008.
- [5] B. Chandrasekaran, “Representing function: Relating functional representation and functional modeling research streams,” *AI EDAM*, vol. 19, no. 02, pp. 65–74, 2005.
- [6] Y. Umeda and T. Tomiyama, “Functional reasoning in design,” *IEEE Expert: Intelligent Systems and Their Applications*, vol. 12, no. 2, pp. 42–48, 1997.

- [7] G. Pahl and W. Beitz, *Engineering Design: A Systematic Approach*. Springer-Verlag London Limited., 2007.
- [8] S. Szykman, J. W. Racz, and R. D. Sriram, “The representation of function in computer-based design,” in *The 1999 ASME Design Engineering Technical Conferences*, September 1999.
- [9] R. B. Stone and K. L. Wood, “Development of a functional basis for design,” *Journal of Mechanical Design*, vol. 122, no. 4, pp. 359–370, 2000.
- [10] J. Hirtz, R. Stone, D. Mcadams, S. Szykman, and K. Wood, “A functional basis for engineering design: Reconciling and evolving previous efforts,” *Research in Engineering Design*, vol. 13, no. 2, pp. 65–82, March 2002.
- [11] Y. Kitamura and R. Mizoguchi, “Ontology-based systematization of functional knowledge,” *Journal of Engineering Design*, vol. 15, pp. 327–351(25), August 2004.
- [12] Y. Kitamura, T. Sano, K. Namba, and R. Mizoguchi, “A functional concept ontology and its application to automatic identification of functional structures,” *Advanced Engineering Informatics*, vol. 16, no. 2, pp. 145 – 163, 2002.
- [13] Y. Kitamura, S. Takafuji, and R. Mizoguchi, “Towards a reference ontology for functional knowledge interoperability,” vol. 2007, no. 48078. ASME, 2007, pp. 111–120.
- [14] H. J. Lee, H. J. Ahn, J. W. Kim, and S. J. Park, “Capturing and reusing knowl-

- edge in engineering change management: A case of automobile development,” *Information Systems Frontiers*, vol. 8, no. 5, pp. 375–394, 2006.
- [15] N. Joshi, “Methodologies for improving product development phases through plm,” Ph.D. dissertation, Mechanical Engineering, University of Michigan, Ann Arbor, MI., 2008.
- [16] R. N. Shepard, “Attention and the metric structure of the stimulus space,” *Journal of Mathematical Psychology*, vol. 1, no. 1, pp. 54 – 87, 1964.
- [17] A. Tversky, “Features of similarity,” in *Psychological Review*, vol. 84, no. 2, 1977, pp. 327–352.
- [18] R. L. Goldstone, “Similarity, interactive activation, and mapping,” *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 20, no. 1, pp. 3–28, 1994.
- [19] B. R. Graedel, T. E. and Allenby, “Matrix approaches to abridged life cycle assessment,” *Environmental science & technology*, vol. 29, 1995.
- [20] V. Kalyanasundaram and B. Bras, “Software tool for evaluation of environmental indicators using cad models early in the design cycle,” in *IDETC/CIE 2009, San Diego, California, USA,, 2009*.
- [21] I. Sousa, D. Wallace, and J. L. Eisenhard, “Approximate life-cycle assessment of product concepts using learning systems,” *Journal of Industrial Ecology*, vol. 4, pp. 61–81, 2000.

- [22] T. R. Gruber, “Toward principles for the design of ontologies used for knowledge sharing,” *International Journal of Human-Computer Studies*, vol. 43, no. 5-6, pp. 907 – 928, 1995.
- [23] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, March 2003.
- [24] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosz, and M. Dean, “Swrl: A semantic web rule language combining owl and ruleml,” W3C Member Submission, World Wide Web Consortium, Tech. Rep., May 2004.
- [25] H. Knublauch, R. W. Ferguson, N. F. Noy, and M. A. Musen, “The protégé owl plugin: An open development environment for semantic web applications,” 2004, pp. 229–243.
- [26] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz, “Pellet: A practical owl-dl reasoner,” *Journal of Web Semantics*, vol. 5, no. 2, pp. 51–53, June 2007.
- [27] H. Eriksson, “Using jesstab to integrate protege and jess,” *Intelligent Systems, IEEE*, vol. 18, no. 2, pp. 43 – 50, mar-apr 2003.
- [28] “Design repository,” Last Accessed January 20, 2009. [Online]. Available: <http://function.basiceng.umr.edu/delabsite/repository.html>
- [29] M. J. O’SConnor, R. Shankar, C. Nyulas, S. Tu, and A. Das, “Developing a web-based application using owl and swrl.” Stanford, CA.: AAAI Spring Symposium, 2008.

- [30] G. Q. Huang, W. Y. Yee, and K. L. Mak, "Development of a web-based system for engineering change management," *Robotics and Computer-Integrated Manufacturing*, vol. 17, no. 3, pp. 255 – 267, 2001.
- [31] I. C. Wright, "A review of research into engineering change management: implications for product design," *Design Studies*, vol. 18, no. 1, pp. 33 – 42, 1997.
- [32] P. Pikosz and J. Malmqvist, "A comparative study of engineering change management in three swedish engineering companies," in *In Proceedings of DETC'98, Atlanta, GA, USA*, 1998.
- [33] R. Keller, C. M. Eckert, and P. J. Clarkson, "Multiple views to support engineering change management for complex products," in *CMV '05: Proceedings of the Coordinated and Multiple Views in Exploratory Visualization*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 33–41.
- [34] SASIG, "Strategic automotive product data standard industry group ecm recommendation," 2009. [Online]. Available: <http://www.sasig.com>
- [35] P. Resnik, "Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language," *Journal of Artificial Intelligence Research*, vol. 11, pp. 95–130, 1999.
- [36] C. Leacock and M. Chodorow, *Combining Local Context and WordNet Similarity for Word Sense Identification*. The MIT Press, May 1998, ch. 11, pp. 265–283.
- [37] W.-S. Li and C. Clifton, "Semint: A tool for identifying attribute correspon-

- dences in heterogeneous databases using neural networks,” *Data Knowl. Eng.*, vol. 33, no. 1, pp. 49–84, 2000.
- [38] D. Lin, “An information-theoretic definition of similarity,” in *In Proceedings of the 15th International Conference on Machine Learning*. Morgan Kaufmann, 1998, pp. 296–304.
- [39] A. K. Jain, M. N. Murty, and P. J. Flynn, “Data clustering: a review,” *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, 1999.
- [40] V. Roth, J. Laub, M. Kawanabe, and J. M. Buhmann, “Optimal cluster preserving embedding of nonmetric proximity data,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 12, pp. 1540–1551, 2003.
- [41] D. W. Jacobs, D. Weinshall, and Y. Gdalyahu, “Classification with nonmetric distances: Image retrieval and class representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 583–600, 2000.
- [42] J. Handl, J. Knowles, and M. Dorigo, “Ant-based clustering and topographic mapping,” *Artif. Life*, vol. 12, no. 1, pp. 35–61, 2006.
- [43] H. A. and Christiane Guinot and G. Venturini, *Ant Colony, Optimization and Swarm Intelligence*. Springer Berlin / Heidelberg, 2004.
- [44] J. Kennedy and R. C. Eberhart, *Swarm intelligence*. Morgan Kaufmann Publishers, 2001.

- [45] J. L. Deneubourg, S. Goss, N. Franks, A. Sendova-Franks, C. Detrain, and L. Chrétien, “The dynamics of collective sorting robot-like ants and ant-like robots,” in *Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats*. Cambridge, MA, USA: MIT Press, 1990, pp. 356–363.
- [46] E. D. Lumer and B. Faieta, “Diversity and adaptation in populations of clustering ants,” in *SAB94: Proceedings of the third international conference on Simulation of adaptive behavior : from animals to animats 3*. Cambridge, MA, USA: MIT Press, 1994, pp. 501–508.
- [47] G. Ameta, S. Rachuri, X. Fiorentini, M. Mani, S. J. Fenves, K. W. Lyons, and R. D. Sriram, “Extending the notion of quality from physical metrology to information and sustainability,” *Journal of Intelligent Manufacturing*, 2009.
- [48] R. A. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1999.
- [49] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data*, 2005.
- [50] A. Y. Ng, M. I. Jordan, and Y. Weiss, “On spectral clustering: Analysis and an algorithm,” in *Advances in Neural Information Processing Systems 14*. MIT Press, 2001, pp. 849–856.
- [51] Y. Avramenko and A. Kraslawski, *Case Based Design: Applications in Process Engineering*. Springer-Verlag Berlin Heidelberg, 2008.

- [52] G. Ameta, M. Mani, S. Rachuri, S. C. Feng, R. D. Sriram, and K. W. Lyons, “Carbon weight analysis for machining operation and allocation for redesign,” *International Journal of Sustainable Engineering*, vol. 2, no. 4, pp. 241–251, 2009.
- [53] Y. S. Song, J. R. Youn, and T. G. Gutowski, “Life cycle energy analysis of fiber-reinforced composites,” *Composites Part A: Applied Science and Manufacturing*, vol. 40, no. 8, pp. 1257 – 1265, 2009, special Issue: 15th French National Conference on Composites - JNC15. [Online]. Available: <http://www.sciencedirect.com/science/article/B6TWN-4WCTWYH-1/2/0fa5c74f587d7fc1ea1590f6c2f8022c>
- [54] M. Schipper, “Energy-related carbon dioxide emissions in u.s. manufacturing,” Department of Energy DOE/EIA-0573, Tech. Rep., 2005.
- [55] Y. Freund and R. E. Schapire, “A decision-theoretic generalization of on-line learning and an application to boosting,” in *EuroCOLT '95: Proceedings of the Second European Conference on Computational Learning Theory*. London, UK: Springer-Verlag, 1995, pp. 23–37.
- [56] B. Efron and R. Tibshirani, *An introduction to the bootstrap*. Chapman & Hall, 1993.