



**HAL**  
open science

# Integrating high-level requirements in optimization problems: theory and applications

Fabio Roda

► **To cite this version:**

Fabio Roda. Integrating high-level requirements in optimization problems: theory and applications. Operations Research [math.OC]. Ecole Polytechnique X, 2013. English. NNT: . pastel-00817782

**HAL Id: pastel-00817782**

**<https://pastel.hal.science/pastel-00817782v1>**

Submitted on 25 Apr 2013

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



# Integrating high-level requirements in optimization problems: theory and applications

Thèse présentée pour obtenir le grade de  
DOCTEUR DE L'ÉCOLE POLYTECHNIQUE

par

Fabio RODA

Soutenue le 01/03/2013 devant le jury composé de:

Leo Liberti	École Polytechnique, Palaiseau (France)	Directeur de thèse
Giorgio Gallo	Università di Pisa, Pisa (Italie)	Rapporteur
Alexis Tsoukiàs	Université Paris Dauphine, Paris 9 (France)	Rapporteur
Olivier Bournez	École Polytechnique, Palaiseau (France)	Examinateur
Franco Raimondi	Middlesex University, London (Royaume-Uni)	Examinateur
Roberto Wolfler Calvo	Université Paris Nord, Paris 13 (France)	Examinateur



*dedicated to Francesca and Sofia*



*“The only way to rectify our reasonings is to make them as tangible as those of the Mathematicians, so that we can find our error at a glance, and when there are disputes among persons, we can simply say: Let us calculate [calculemus], without further ado, to see who is right.”<sup>1</sup>*

---

<sup>1</sup>G. W. Leibniz, *The Art of Discovery*, 1685, in *Leibniz: Selections*. Edited by Philip P. Wiener. New York: Charles Scribner’s Sons, 1951.



# Abstract

The development of a system is always motivated by desiderata that are expressed by customers, end-users and other stakeholders who are external to the system to be realized. In order to manage the design process of a system in conformity with stakeholders' needs several special techniques are necessary. Some disciplines deal with this problem. One of these, Systems Engineering (SE), emerged in 1950s to help design and management of complex systems (namely systems with a high number of heterogeneous components and interactions) and proposed a suitable methodology. It adopts an *holistic* and *top-down* approach. SE, considering the system and its environment as a whole, addresses the main issues globally, in the initial phases of the system design process. Then, progressively breaking down the system into smaller components, systems engineers go down into details and identify interactions and interfaces between components. The first task of *systems design* is to identify and organize the stakeholders' needs and to specify them in a clear set of requirements, namely a set of non ambiguous, testable and measurable features that the system should have. Requirements can have different levels of abstraction. Initially they concern the entire system. They are referred as system-level or high-level requirements. During the system design process, requirements are progressively refined and associated to sub-systems and components. At each phase of this process, modelling techniques are employed in order to provide prototypes that can help to refine the design.

*Mathematical Programming* (MP) is the branch of Operation Research (OR) that provides a general formal language useful to describe and solve optimization problems.

We mix MP and SE modelling methods integrating system-level requirements in optimization problems. We use MP, in the first phases of the design process, as a common methodology to model prototypes that fulfil high-level requirements, consistently with the holistic, top-down approach of SE. This approach is applied to three different kinds of system.

*Information systems*, i.e. the network of communication channels, hardware,



software and trained people used within an enterprise to help planning and control, are required to provide a basis for numerous IT projects that are launched in order to respond to the needs of the business. Thus, they shall be able to evolve in its entirety due to the replacement of an existing software technology by a new one (e.g. passing from several independent/legacy software packages to an integrated one, migrating from an existing IT technology to a new one, and so on). These evolutions (or *transitions*) invariably have a strong impact at the IT layer level, where existing IT modules are replaced by new ones. This translates to a replacement of existing services by new services ensuring that the impact on the whole enterprise is kept low in order to avoid business discontinuity. Such an objective leads thus typically to the necessity of co-optimizing both creation *and* replacement/destruction, called usually *kills* in the IT language, of parts of the information system, and of prioritizing the IT responses to the business consequently. The information system shall ensure profitable services and it shall be maintainable and extensible. We propose an operational model and a mathematical programming formulation expressing a generic global prioritization problem occurring in the (limited, but practically relevant) context of a technological evolution of an information system.

*Recommender systems* (RS) are a kind of search engine and aim to provide personalized recommendations. The amount of available sources of information has strongly increased during the last decades. The overwhelming growth of the Web is probably the clearer example. From one hand this is a wonderful progress but on the other hand this poses problems, because it is not easy to retrieve the relevant information when required. Too much information can puzzle users and decision makers. Thus, the need for new technologies that can help search and retrieve information is huge. Recommender systems (RS) are one of these technologies. RS are designed to help users, who lack knowledge to evaluate the high number of alternatives provided by several sources, first of all the Web. We consider the problem of designing recommender systems in order to provide good, interesting and accurate suggestions. We propose three recommender systems: TMW, which is combinatorial optimization based, BMC, which is clusters modularity based and LSPR, which exploits information retrieval techniques. We use precise metrics (accuracy, audacity, computational efficiency) and test the systems in order to identify the best one.

The transportation of hazardous materials (hazmat) entails several issues due to the environmental consequences of possible accidents. We can figure it this way: there are some trucks that have to transport some kind of dangerous material from one or many production points to one or many garbage dumps, crossing different areas. The concerned *transportation system* includes road network, garbage sources, waste disposal plants and is influenced by several stakeholders as national government agencies and political jurisdictions, local governments, transportation industry, the media, public sector interest groups or trade associations, public health and en-

vironmental interest groups. The transportation system shall ensure safe disposal of hazardous waste in such a way that the risk of potential catastrophic accident is equitably distributed over the population. We have to select a set of paths which is optimal from the point of view of cost, risk and equity. Equity is somehow unusual and is hard to define. We consider and integrate in MP formulations two different ideas of equity. The first approach simply requires that all areas involved in the transportation network share the same level of risk. The second definition of equity is inspired by the concept of “*equity as fairness*” of J. Rawls. We evaluate the relationships between equity requirements and global risk.



# Résumé

Le développement d'un système est toujours motivé par des besoins qui sont exprimés par des clients, des utilisateurs et d'autres parties prenantes externes au système qu'il faut réaliser. Pour maîtriser le processus de développement du système conformément aux besoins des parties prenantes, plusieurs techniques spéciales sont nécessaires. Certaines disciplines s'occupent de ce problème. Une de ces disciplines, l'Ingénierie Système, est née/apparue dans les années 50 pour aider le design et la gestion des systèmes complexes (c.à.d. systèmes qui ont un nombre considérable de composants hétérogènes qui interagissent mutuellement). Elle a proposé une méthodologie adaptée. Elle adopte une méthode qui est holistique et descendante (top-down). L'Ingénierie Système, en considérant le système et son environnement comme une unité, considère les problèmes majeurs dans les premières phases du processus du développement du système. Ensuite, en décomposant le système en parties plus petites, les ingénieurs système rentrent dans les détails et identifient les interactions et les interfaces entre les composants. Le premier but de la phase de conception des systèmes est d'organiser les besoins des parties prenantes dans un ensemble d'exigences claires, c.à.d. un ensemble de caractéristiques non ambiguës, testables et mesurables que le système doit avoir. Ces exigences peuvent présenter différents niveaux d'abstraction. Au début, les exigences regardent le système globalement. Elles sont appelées exigences de haut niveau. Pendant le processus de design, elles sont progressivement affinées et affectées aux sous-systèmes et composants. Pour chacune des phases de ce processus, des techniques de modélisation sont utilisées pour pouvoir produire des prototypes qui peuvent aider à l'affinement du design. La programmation mathématique est un secteur de la recherche opérationnelle qui fournit un langage formel général qui est utile à décrire et à résoudre les problèmes d'optimisation. Nous utilisons, ensemble, l'ingénierie système et la programmation mathématique pour intégrer les exigences de haut niveau dans des problèmes d'optimisation. Nous utilisons la programmation mathématique dans les premières phases du processus de la conception du système comme trait-d'union pour modéliser des prototypes qui satisfont les exigences de haut niveau, d'une façon cohérente avec la méthode holistique et descendante (top-down) de l'ingénierie système. Nous appli-

quons cette méthode à trois types différents de système.

Les Systèmes d'Information (SI), c.à.d. les réseaux des ressources, matérielles, logicielles et utilisateurs, utilisés dans une entreprise pour aider au contrôle et à la planification, doivent fournir la base des projets qui sont lancés pour répondre aux besoins commerciaux/des affaires (business). Les SI doivent être capables d'évoluer au fil des remplacements d'une technologie par une autre (e.g. le passage des différents logiciels indépendants à un unique, intégré, ou d'une version à une autre, et cetera). Ces évolutions induisent de grandes conséquences au niveau de la couche IT, où de vieux modules sont remplacés par des nouveaux. Il faut consentir le remplacement des services existants par de nouveaux services en minimisant les perturbations pour les utilisateurs, pour limiter le risque de discontinuité commerciale (business).

Cet objectif correspond à la nécessité d'optimiser à la fois le remplacement/-destruction d'une partie du système d'information et à prioriser les réponses de la division IT au business.

Le système d'information doit garantir des services rentables, et doit être maintenable et extensible. Nous proposons un modèle opérationnel et une formulation de programmation mathématique qui formalise un problème de priorisation qui se présente dans le contexte (limité mais en pratique pertinent) de l'évolution technologique d'un système d'information.

Les Recommender Systems (RS) sont un type de moteur de recherche dont l'objectif est de fournir des recommandations personnalisées. La disponibilité des sources d'informations a remarquablement augmenté ces dernières années. La gigantesque croissance du Web nous en donne, probablement, l'exemple le plus clair. Même si cela relève d'une évolution formidable, néanmoins elle présente une difficulté à repérer l'information appropriée quand recherchée/nécessaire. L'excès d'informations peut perdre/dérouter utilisateurs et décideurs. Aussi apparaît un grand besoin de nouvelles technologies qui peuvent aider à chercher et à repérer l'information. Les Recommender Systems (RS) sont une de ces technologies. Les RS sont conçus pour aider les utilisateurs, qui manquent de connaissances, à évaluer de grands nombres d'alternatives proposées par différentes sources, surtout de la part du Web. Nous considérons le problème du design des Recommender Systems dans le but de fournir de bonnes, intéressantes et précises suggestions. Nous proposons trois Recommender Systems : TMW, qui est basé sur l'optimisation combinatoire ; BMC, qui est basé sur le clustering et la modularité ; et LSPR, qui exploite des techniques de Recherche d'Information (RI). Nous utilisons des mesures spécifiques (*accuracy*, *audacity*, rapidité de calcul) pour tester les systèmes et en identifier le meilleur.

Le transport des matériaux dangereux entraîne plusieurs problèmes liés aux conséquences écologiques des incidents possibles. Nous pouvons décrire ce scénario de la façon suivante : des camions doivent transporter des matériaux dangereux à partir d'un ou plusieurs centres de production vers un ou plusieurs points de décharge, à travers des différentes régions. Le système comprend routes, centres de

production des déchets, points de décharge. Il est conditionné par plusieurs parties prenantes comme le gouvernement national, les institutions politiques, les autorités locales, l'industrie du transport, les médias, les groupes d'intérêt public, les associations de commerce et de défense écologique. Le système de transport doit assurer le transport, pour l'élimination en sécurité des déchets dangereux, d'une façon telle que le risque des possibles incidents soit distribué d'une manière équitable parmi la population. Nous devons sélectionner un ensemble de parcours qui est optimal du point de vue des coûts, des risques et de l'équité. L'équité est une notion plutôt inhabituelle dans notre domaine, et elle est difficile à définir. Nous considérons et intégrons dans des formulations de programmation mathématique deux idées différentes d'équité. La première approche demande simplement que toutes les régions impliquées dans le transport obtiennent le même niveau de risque. La seconde définition d'équité s'inspire du concept de *fairness* de J. Rawls. Nous évaluons les relations entre équité et risque.



# Acknowledgements

I wish to thank my supervisor, Leo Liberti for his help. He has taught me that integer numbers can be as interesting as trasfinite ones.

Daniel Krob and Cesames that supported my work.

Gilles Barbier and Dismoioù for the initial chance they gave me.

I wish to thank Francesca and my family for their support and love.

My daughter Sofia for inspiring my days (and sleepless nights . . .).

Alberto, Chiara C., Chiara L., Franco and Xavier for their friendship and help.

Myself, for my perseverance.





# Contents

<b>I</b>	<b>Theory</b>	<b>1</b>
<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Overview . . . . .	3
1.2	Mathematical Programming . . . . .	4
1.2.1	Multiobjective Mathematical Programming . . . . .	6
1.2.2	The meaning of “minimization” in MOMP . . . . .	8
1.2.3	Solution methods . . . . .	10
1.2.4	Computational complexity . . . . .	12
1.3	Systems Engineering . . . . .	14
1.3.1	Requirements engineering . . . . .	14
1.3.2	Goals . . . . .	19
1.3.2.1	Tropos . . . . .	20
1.3.3	Systems design and management process . . . . .	22
1.3.4	From goals to objective functions . . . . .	24
1.4	Structure of the work . . . . .	25
<b>II</b>	<b>Applications</b>	<b>29</b>
<b>2</b>	<b>The information system architecture evolution problem</b>	<b>31</b>
2.1	Introduction . . . . .	31
2.2	Operational model of an evolving information system . . . . .	32
2.2.1	Elements of information system architecture . . . . .	32
2.2.2	Global Goal . . . . .	32
2.2.3	Evolution of an information system architecture . . . . .	33
2.2.4	Management of IT system architecture evolutions . . . . .	34
2.2.5	The IT system architecture evolution management . . . . .	35
2.3	Mathematical Programming based approach . . . . .	35
2.3.1	Introducing the MP formulation . . . . .	37
2.3.2	Sets, variables, objectives, constraints . . . . .	38

2.3.3	Valid cuts from implied properties . . . . .	42
2.4	Formulation properties and trade-off . . . . .	44
2.4.1	Decomposability . . . . .	44
2.4.2	Trade-off . . . . .	46
2.5	Computational results . . . . .	49
2.5.1	CPU time . . . . .	50
2.5.2	Optimality Gap . . . . .	52
2.5.3	Cuts effectiveness . . . . .	53
2.5.4	Trade-off in realistic instances . . . . .	53
<b>3</b>	<b>The recommendation problem</b>	<b>57</b>
3.1	Introduction . . . . .	57
3.1.1	Recommendation problem . . . . .	59
3.1.2	Recommendation methods . . . . .	59
3.2	Operational model of a recommender system . . . . .	61
3.2.1	Global Goal . . . . .	62
3.3	Recommender system 1: TMW . . . . .	63
3.3.1	Formal model . . . . .	63
3.3.2	Identification of maximum confidence paths . . . . .	65
3.3.3	Exploiting the ratings . . . . .	66
3.4	Recommender system 2: BMC . . . . .	67
3.4.1	Bipartite networks and modularity . . . . .	68
3.4.2	RS by means of bipartite modularity based clustering . . . . .	69
3.5	Recommender system 3: LSPR . . . . .	70
3.5.1	Model . . . . .	70
3.6	Evaluating the recommender systems . . . . .	73
3.6.1	Dataset . . . . .	73
3.6.2	Evaluating effectiveness . . . . .	73
3.6.2.1	Accuracy . . . . .	73
3.6.2.2	Audacity . . . . .	75
3.6.3	Evaluating efficiency . . . . .	76
3.7	Recommender systems design problem . . . . .	77
3.8	Evaluation TMW using a small dataset . . . . .	78
<b>4</b>	<b>The equitable hazmat transportation problem</b>	<b>81</b>
4.1	Introduction . . . . .	81
4.2	Related works . . . . .	82
4.3	Operational model of a transportation system . . . . .	84
4.3.1	Global Goal . . . . .	84
4.4	Mathematical programming formulation . . . . .	86
4.4.1	Sets, variables, objectives, constraints . . . . .	87

4.4.2	Reformulation of the model . . . . .	89
4.4.2.1	$\epsilon$ -constraint method . . . . .	90
4.5	Computational results . . . . .	92
4.5.1	Preliminary tests . . . . .	92
4.5.2	Comparison . . . . .	93
4.6	A heuristic for large-scale instances . . . . .	95
4.6.1	Graph reduction through centrality erosion . . . . .	96
<b>III</b>	<b>Meta-theory</b>	<b>101</b>
<b>5</b>	<b>Some epistemological and historical remarks</b>	<b>103</b>
5.1	Philosophy of Engineering . . . . .	103
5.1.1	Epistemic vs non-epistemic values . . . . .	106
5.2	Models . . . . .	107
5.2.1	Model validation . . . . .	108
5.3	Experimental approach to scientific knowledge . . . . .	109
5.4	Collaborative approach to scientific knowledge . . . . .	113
5.5	Ethical approach to scientific knowledge . . . . .	115
5.6	Computational approach to scientific knowledge . . . . .	118
5.7	Teleological approach to scientific knowledge . . . . .	121
<b>IV</b>	<b>Conclusions</b>	<b>123</b>
<b>6</b>	<b>Conclusions</b>	<b>125</b>
<b>V</b>	<b>Bibliography and appendices</b>	<b>129</b>
	<b>Bibliography</b>	<b>131</b>
	<b>Appendices</b>	<b>145</b>
<b>A</b>	<b>Publications</b>	<b>147</b>



**Part I**

**Theory**



# Chapter 1

## Introduction

### 1.1 Overview

In this work we focus on systems design. Systems are realized to fulfil the needs of a set of stakeholders<sup>1</sup>. In order to achieve this goal, system designers have to define a set of requirements, i.e. non ambiguous, verifiable and measurable features that the system should have. When a system is designed from scratch, first of all, a set of high-level requirements, that concern the system as a whole, are settled. These high-level requirements define the aims of the system. High-level behavioral requirements define functions and services that represent the core activities required by the users. High-level quality (i.e. non-behavioral) requirements define auxiliary features that determine some of the system's qualities (for example, portability and accessibility).

Even if evidence shows that incorrect assumptions about the motivations of a project can lead to a very poor outcome, several requirements engineering approaches do not pay special attention to the environment of the system and to stakeholders' needs. Understanding *why* the system is needed is not always a fully considered question, the focus being more on *what* it shall do and *how*. In practice, this means that the phase of high-level requirements elicitation and analysis, is often not sufficiently accurate. In general, at this stage, behavioral requirements are better defined than quality ones. From a purely technical point of view, this depends on the fact that non-behavioral requirements are often expressed in blurred and/or informal terms. However, non-behavioral requirements have an important role in ensuring the realization of high quality systems.

Typically, the construction of a system goes through the conception of intermediate, abstract simplified models that help system designers take their decisions. An important phase in this process concerns the optimization of these choices, namely

---

<sup>1</sup>A stakeholder is an external entity, not necessarily a person, that affects the system or is affected by the system.



the selection of the best ones among all admissible ones. Mathematical programming (MP) is a formal language used to describe and solve optimization problems. The optimization of a system's features can thus be based on the formulation of a MP model that describes the system's requirements. Nevertheless, normally, optimization does not intervene in the very early stages of the systems design process. The first abstract models, which are based on high-level requirements, are not written using MP formulations. MP formulations are typically reserved for detailed models based on component-level requirements. On the contrary, we anticipate and extend the use of optimization methods in the system design cycle. In particular, we try to improve the very early phases of design, corresponding to high-level requirements definition. We integrate quality requirements in MP formulation to make them precise and better identify real trade-offs.

The first interpretation of this thesis is simply a collection of case studies for mathematical programming. Nevertheless, we think there is more to this work than a simple set of examples. The *trait d'union* is represented, as pointed out above, by the enforcement of non-behavioral requirements in our MP formulations, which, as already mentioned, are technically difficult to treat quantitatively. In general, we deal with the problem of integrating high-level requirements in MP formulations in order to improve the early phases of systems design. In particular, we apply this approach to three different types of system: information systems, recommender systems and transportation systems. We focus on three different specific problems, in each of these systems: the information system architecture evolution problem, the recommendation problem and the equitable hazardous materials transportation problem. They are introduced in dedicated chapters where we present context, motivations, formulations and methods. In the next sections we would like to supply the reader with some general topics and underline some background context that is common to all the problems we present in the following chapters. In Section 1.2 we provide introductory elements of mathematical programming and in Section 1.3 of systems engineering.

## 1.2 Mathematical Programming

Traditionally, *Mathematical Programming* (MP) is defined as the branch of Operation Research (OR)<sup>2</sup> that is dedicated to solve optimization problems.

*“What does mathematical programming mean? Programming here means “planning”. Literally, [...] “mathematical models for planning”.”* [142]

---

<sup>2</sup>Operations research (OR) is “a scientific approach to analyzing problems and making decisions” ([http://www.hsor.org/what\\_is\\_or.cfm](http://www.hsor.org/what_is_or.cfm)) and/or “the application of scientific methods to the management and administration of military, government, commercial, and industrial systems” (<http://global.britannica.com/>).

More recently, Liberti [105] proposes to define MP as a formal language: “. . . *there is a formal language that can be used to describe [problems]: mathematical programming (MP)*”. In this work, we adopt this point of view.

MP provides us with a common method to formalize and solve the problems presented in the following chapters.

Formally we can define a *mathematical program* as follows.

$$\left. \begin{array}{l} \min \quad f(x) \\ \text{s.t.} \quad \forall i \quad 1 \leq i \leq m \quad g_i(x) \geq 0, \\ \quad \quad \forall j \quad 1 \leq j \leq n \quad h_j(x) = 0, \\ \quad \quad x \in X, \end{array} \right\} \quad (1.1)$$

where:

- $f : X \rightarrow \mathbb{R}$  is the *objective function*,
- $X$  is the domain for the decision variables  $x$ ,
- For each  $i$ ,  $1 \leq i \leq m$ ,  $g_i : X \rightarrow \mathbb{R}$  is the left hand side of the  $i$ -th inequality constraint,
- For each  $j$ ,  $1 \leq j \leq n$ ,  $h_j : X \rightarrow \mathbb{R}$  is the left hand side of the  $j$ -th equality constraint.

The interpretation of model (1.1) could be paraphrased as: “*find a point  $x \in X$  which satisfies both equality and inequality constraints, and which minimizes the objective function  $f(x)$* ”. If it exists, this point is called *optimal solution*. If it does not exist, we say that the problem is *infeasible*.

According to [84], MP as a mathematical discipline includes any or all of the following:

- Theorems about the form of a solution, including whether one exists;
- Algorithms to seek a solution or ascertain that none exists;
- Formal problem descriptions as MP formulations;
- Analysis of the formulation properties, including infeasibility or unboundedness;
- Theorems about the model structure, including properties pertaining to feasibility, redundancy and/or implied relations (such theorems are often used to design more efficient solution algorithms);
- Theorems about solution optimality: whether it is local or global, whether it comes with an approximation guarantee

There are several type of MP, depending on the form of the objective, the constraints and the domain of decision variables. The following is a simple but common classification:

- Linear Programming (LP): the objective function and the constraints are linear, and the variables are continuous;
- Mixed-Integer Linear Programming (MILP or MIP): the objective function and the constraints are linear, and some variables might be constrained to take integer values;
- convex Nonlinear Programming (cNLP): the objective function and the constraints are convex, some of them might involve nonlinear terms, and the variables are continuous;
- Nonlinear Programming (NLP): at least one among the objective function and the constraints might be nonlinear, and the variables are continuous;
- convex Mixed Integer Nonlinear Programming (cMINLP): the objective function and the constraints are convex, some of them might be nonlinear, and some variables might be constrained to take integer values;
- Mixed Integer Nonlinear Programming (MINLP): at least one among the objective function and the constraints might be nonlinear, and some variables might be constrained to take integer values.

### 1.2.1 Multiobjective Mathematical Programming

The problems we consider in this work have more than one objective, as we will see in the next sections. They belong to the class of multiobjective problems. In order to provide a taxonomy of multiobjective decision making approaches, we introduce some specifications. A first important distinction is between the following two scenarios. In the first case, the possible choices are explicitly known from the beginning, and the decision maker has to choose among them. In the second scenario the set of possible choices is not immediately available since they are expressed implicitly as the solutions of a mathematical model. A second distinction is based on the number of possible choices, which can be finite or (potentially) infinite. These two features are often coupled, thus we have two main contexts:

- Scenario 1: a finite set of explicitly known choices
- Scenario 2: a (potentially) infinite set of implicitly known choices.

In the first case the focus is on choice selection, and in the second one it is on choice identification. These two tasks attracted different communities of researchers and there is by now a quite clear separation between those who focus on finding the

set of possible solutions and those who focus on aiding to make the choice among all possible ones.

*Multi Criteria Decision Analysis* (MCDA) and *Multi Objective Mathematical Programming* (MOMP) are used respectively for the finite-explicit scenario and for the infinite-implicit one. *Multi Criteria Decision Making* (MCDM) is a most general term, which refers to both of them.

*“MCDM can be classified into two main branches of multicriteria decision analysis (MCDA) and multiobjective Mathematical programming (MOMP). The former applies mainly when there are a small number of actions, the latter when the number of actions is large. Usually, MCDA applies to decision problems with discrete actions, and MOMP when the action space is continuous”.* [152]

*“we understand multi objective programming as pertaining to situations where feasible alternatives are available implicitly, through constraints in the form of mathematical functions. An optimization problem (typically a mathematical program) has to be solved to explicitly find the alternatives. Decision problems with multiple criteria and explicitly available alternatives are treated within multicriteria decision analysis (MCDA). This view constitutes the difference between multi objective programming and multicriteria decision analysis (MCDA) which complement each other within multicriteria decision making (MCDM).”* [59]

According to [59] the conference on “Multiple Criteria Decision Making” (MCDM) organized in the 1972 at Columbia University in South Carolina opens officially the field, but, actually, its roots date back into a more distant past.

*“The earliest known reference relating to Multiple Criteria Decision Making (although not using that name) can be traced to Benjamin Franklin (1706-1790), the American statesman, who allegedly had a simple paper system for deciding his position on an important issue.”* [93]

The terminology about multiobjectives decision making is not totally uniform in literature. Nevertheless, in general, context helps to understand if we deal with a scenario of type 1 or of type 2, so that confusion is avoided. Among others, alternative terms for Scenario 1 are *Multiattribute Decision Making Problems* (MADM) and *Multiple Criteria Evaluation Problems* and alternative terms for Scenario 2 are *Selection Problems* and *Multiple Criteria Design Problem*. *Multiobjective optimization* (MOP) is another widespread, general term.

Another distinction is relevant. Some authors (for example see [167, 168]) underline the difference between *understanding the decision-making process, aiding to decide* and *actual decision making*. In fact, they remark that we might want:

- to study theoretically the decision making process
- to help decision makers to produce a decision
- to provide a method for effective decision making.

We focus on MOMP. In presence of two or more objectives the model 1.1 extends quite naturally. A general multiobjective mathematical problem is defined as follows:

$$\left. \begin{array}{l} \min \quad F(x) = [f_1(x), f_2(x), \dots, f_{k^*}(x)]^T \\ \text{s.t.} \quad \forall i \quad 1 \leq i \leq m \quad g_i(x) \geq 0, \\ \quad \quad \forall j \quad 1 \leq j \leq n \quad h_j(x) = 0, \\ \quad \quad x \in X, \end{array} \right\} \quad (1.2)$$

where:

- $F(x) : X \rightarrow \mathbb{R}^p$  is a vector of  $k^*$  *objective functions*  $f_k(x)$  which are called *criteria* (also called payoff functions, cost functions, or value functions).
- $x \in X$  is a vector of *decision variables* (also called design variables)
- The *feasible set*  $X$  (also called design space) is defined as the set  $\{x \mid \forall i \quad 1 \leq i \leq m \quad g_i(x) \geq 0 \text{ and } \forall j \quad 1 \leq j \leq n \quad h_j(x) = 0\}$ .
- The *criterion space*  $Z$  (also called the feasible cost space or the attainable set) is defined as the set  $\{F(x) \mid x \in X\}$ .

### 1.2.2 The meaning of “minimization” in MOMP

There is no standard total order on  $\mathbb{R}^p$ , thus the meaning of “minimization” in equation 1.2 requires a clarification, since points in the criterion space  $Z$  can be compared in different ways. A possible interpretation is based on two fundamental concepts in MOP, *Pareto optimality* and *nondominance*.

**Definition 1.2.1 (Pareto optimality)** *A feasible solution,  $x^* \in X$ , is Pareto optimal iff there is no other feasible solution  $x \in X$ , such that  $f_i(x) \leq f_i(x^*)$  for all  $i \leq k$ , and  $f_i(x) < f_i(x^*)$  for at least one  $i \leq k$ .*

Alternative terms for referring to Pareto optimal points are Edgeworth-Pareto optimal points and efficient points. The set of all Pareto optimal points is called Pareto efficient frontier.

**Definition 1.2.2 (Weakly Pareto optimality)** *A feasible solution,  $x^* \in X$ , is weakly Pareto optimal iff there does not exist another feasible solution,  $x^* \in X$ , such that  $f_i(x) < f_i(x^*)$  for all  $i \leq k$ .*

Pareto optimal points are weakly Pareto optimal, but weakly Pareto optimal points are not Pareto optimal.

**Definition 1.2.3 (Dominance)** A vector of objective functions,  $F(x^*) \in Z$ , is nondominated iff there does not exist another vector,  $F(x) \in Z$ , such that  $f_i(x) \leq f_i(x^*)$  with at least one  $f_i(x) < f_i(x^*)$ . Otherwise,  $F(x^*)$  is dominated.

We underline the fact that Pareto optimality concerns points in the decision space and that dominance concerns points in the criterion space. According to [55] “we have to remark that these notations are not unique in literature” and that some authors do not distinguish between terms as *efficient solution*, *Pareto optimum* and *nondominated point*. Table 1.1 (that is based on [55], where there is discussion on this topic) shows some examples of this not uniform use of terminology. This is uncomfortable, however, normally, the meaning becomes clear from the context and these terms can be used synonymously with no misunderstanding.

Author	Decision space	Objective space
Sawaragi et al. (1985)	efficient solution	efficient element
Chankong and Haimes (1983)	noninferior solution	noninferior solution
Yu (1985)	Pareto optimal point	Pareto optimal outcome
Miettien (1999)	Pareto optimal decision vector	Pareto optimal criterion vector
Deb (2001)	Pareto optimal solution	Pareto optimal solution
Jahn (2004)	Edgeworth-Pareto optimal point	minimal element

Table 1.1: Not uniform terminology for Pareto optimality and nondominance

Pareto optimality and dominance provide the elements to define a possible precise interpretation of the problem represented by Equation 1.2. We look for nondominated points. This is the meaning of minimization, in MOMP.

Nevertheless, this is not the only possible approach. For example, alternative ones are, among others, the lexicographical and the min-max approaches. Lexicographic optimization is used when we can set an absolute ranking of the criteria. Following [55] a feasible solution,  $x^* \in X$ , is *lexicographically optimal* if there is no other feasible solution  $x \in X$ , such that  $f(x) <_{lex} f(x^*)$  (where  $y^1 <_{lex} y^2$  if  $y_q^1 < y_q^2$  and  $q = \min \{k : y_k^1 \neq y_k^2\}$ ).

In this case, we want to solve the following problem.

$$\left. \begin{array}{l} \text{lex min } F(x) = [f_1(x), f_2(x), \dots, f_{k^*}(x)]^T \\ \text{s.t. } \quad \forall i \quad 1 \leq i \leq m \quad g_i(x) \geq 0, \\ \quad \quad \forall j \quad 1 \leq j \leq n \quad h_j(x) = 0, \\ \quad \quad x \in X, \end{array} \right\} \quad (1.3)$$

In other words, we consider the optimization of  $f_k(x)$  only if optimality for  $f_1(x), f_2(x) \dots f_{k-1}(x)$  have been achieved.

In other situations we want to minimize the worst (i.e. the highest, if we are minimizing) among all criteria. In this case, we want to solve the following problem.

$$\left. \begin{array}{l} \min \max_{i \leq k} f_i(x) \\ \text{s.t.} \quad \forall i \quad 1 \leq i \leq m \quad g_i(x) \geq 0, \\ \quad \quad \forall j \quad 1 \leq j \leq n \quad h_j(x) = 0, \\ \quad \quad x \in X, \end{array} \right\} \quad (1.4)$$

In general, the objective function vectors of the objective space are mapped from  $\mathbb{R}^p$  into an ordered space, e.g.  $(\mathbb{R}^p, \leq)$ , through a *model map*  $\theta$ . The order relation  $\leq$  makes the comparison possible.

*“Feasible set, objective function vector  $f$ , and objective space are data of the [MOMP]. The model map provides the link between objective space and ordered set, in which, finally the meaning of the minimization is defined.*

*Thus, with the three main aspects data, model map, and ordered set the classification  $(X, f, \mathbb{R}^p)/\theta/(\mathbb{R}^p, \leq)$  completely describe a multicriteria optimization.” [55]*

### 1.2.3 Solution methods

As mentioned above, in some cases (Scenario 1) the feasible set is known at the outset, and we “only” have to help the decision maker to make his/her choice. In some other cases (Scenario 2) we have (a) to look for the Pareto optimal set and (b) to aid the decision maker in his/her last choice. If so, the solution process for a MOMP includes a first phase, which involves the computation of all efficient solutions, and a second one, which involves the selection of the most desirable efficient solution with respect to a set of preferences given by the decision maker.

In the context of multiobjective problems we shall consider the preferences of the decision maker about the ranking of the objectives. As suggested in [60], we might say that there are two types of constraints in multiobjective decision making:

*“Domain constraints [that] express the domain of definition of the objective function, [and] preference constraints [that impose] further restrictions on the solution of the problem according to knowledge at a higher level.”*

The role played by decision maker’s preferences determines four approaches, that are referred in literature as follows: *no-preference*, *“a posteriori”*, *“a priori”* and *interactive* (cf. [120]).

1. *No-preference* methods do not use preferences and propose only one solution. In this case, the two phases are not clearly distinct; the final choice is based on a user-independent criterion.
2. “*A priori*” methods make a limited use of preferences during the optimization process. Specifically, preferences are used to limit the extent of the efficient set found in the first phase.
3. “*A posteriori*” methods search the whole efficient set and use preferences to delimit the final output given in the second phase. Typically, these methods are employed when we want to emphasize the presence of a trade-off rather than to provide a “small” solution set.
4. *Interactive* methods mix different aspects of methods from previous categories.

There are several solutions methods available for MCDM. The following (not exhaustive) list shows some of them. In the next chapters, we present in details the methods we need, when we use them (see [59, 55] for a comprehensive exposition).

- Cost-benefit analysis
- Maximin and maximax methods
- Conjunctive and disjunctive methods
- Lexicographic method
- MAUT methods
- Analytic Hierarchy Process (AHP)
- Outranking methods (ELECTRE, PROMETHEE)
- Weighted sum method
- Adaptive Weighted Sum
- $\epsilon$ -constraint method
- Normal Boundary Intersection
- Elastic Constraint method
- Benson’s Method
- Compromise solutions



### 1.2.4 Computational complexity

In order to solve mathematical programs, we need algorithms. One of the tasks of OR is the finding of the most efficient algorithm that solves a given problem. In this section we focus on the concept of computational efficiency and we remind some elements of *computation complexity theory* (see [134,141], on which this section is strongly based, and [69]).

A decision problem is a problem for which the possible answers are “yes” or “no”. Computation complexity theory classifies the algorithms that solve a given decision problem, depending on their complexity. In order to achieve this task, algorithms have to be described in a uniform way, abstracting several details.

Turing Machines (TM) are the standard computation model that is used to evaluate the complexity of the algorithms employed in decision problems. We can figure out a Turing machine as a tape on which a cursor reads, writes or overwrites symbols moving to assigned positions. Formally, a Turing machine is a tuple  $M = (S, \Sigma, \delta, s)$ , where:

- $S$  is a finite set of states;
- $\Sigma$  is a finite set of symbols, called the *alphabet* of  $M$ . The set  $\Sigma$  includes the special symbols  $\sqcup$  and  $\triangleright$ , denoting a blank symbol and the “first” symbol;
- $\delta : S \times \Sigma \rightarrow (S \cup \{\text{yes, no, } h\}) \times \Sigma \times \{\leftarrow, \rightarrow, -\}$ , where  $\{\text{yes, no, } h\}$  is the set of the special *halting* states of  $M$  and  $\{\leftarrow, \rightarrow, -\}$  is the set that includes the symbols denoting cursor directions, is a transition function. We assume that  $S$ ,  $\Sigma$ ,  $\{\text{yes, no, } h\}$  and  $\{\leftarrow, \rightarrow, -\}$  are disjoint sets. The function  $\delta$  is also referred as the *program* of the machine;
- $s \in S$  is the initial state of  $M$ .

$\Sigma^*$  is the set of all the strings composed by symbols of  $\Sigma$  (excepted the null string). A *configuration* of a Turing machine  $M$  is a tuple  $(s, \sigma, \sigma')$ , where  $s \in S$  is a state, and  $\sigma, \sigma' \in \Sigma^*$  are strings.  $\sigma$  is the string to the left of the cursor (including the symbol at the cursor’s position), and  $\sigma'$  is the string to the right.

The input of a TM is a string  $x \in (\Sigma \setminus \sqcup)^*$ . We say that the TM  $M$  accepts a given input, if, for that input, it halts in the “yes” halting state. Formally, we refer to this case with the expression  $(M(x) = \text{yes})$ . We say that the TM  $M$  rejects the input if it halts in the “no” halting state. Formally,  $(M(x) = \text{no})$ .

A language  $L \subseteq \Sigma^*$  is decided by a TM  $M$  if, for all strings  $x \in L$ ,  $M(x) = \text{yes}$  and, for all strings  $x \notin L$ ,  $M(x) = \text{no}$ . If a language  $L$  is decided by a TM, then  $L$  is said to be *recursive*. A language  $L \subseteq \Sigma^*$  is accepted by a TM  $M$  if, for all strings  $x \in L$ ,  $M(x) = \text{yes}$ . The term *recursive enumerable* is used to qualify a language that is accepted by a TM. Recursive enumerability is weaker than recursion. In fact, remark that it is not required that the machine halts when  $x \notin L$ .

TM can be extended to multi-string TM: a  $k$ -string TM (where  $k \geq 1$  is an integer) is a tuple  $M = (S, \Sigma, \delta, s)$ , where  $S$  and  $\Sigma$  are as above, and the transition function  $\delta$  takes into account the  $k$  strings of  $M$ . Formally,  $\delta : S \times \Sigma^k \rightarrow (S \cup \{\text{yes, no, h}\}) \times (\Sigma \times \{\leftarrow, \rightarrow, -\})^k$ . A configuration of a  $k$ -string TM is a  $(2k+1)$  tuple  $(s, \sigma_1, \sigma'_1, \dots, \sigma_k, \sigma'_k)$ , defined as for single tape Turing machines. The program  $\delta$  define the next symbol and the next move of the cursor for each string. The output of a  $k$ -string TM is represented by the last string.

The number of steps necessary to a (multi-string) TM  $M$  on input  $x$  to reach the halting state is called time required to halt. If for any string  $x \in \Sigma^*$ , the time required by a TM  $M$  on input  $x$  is at most  $f(|x|)$ , where  $f(n)$  is a function  $f : \mathbb{N} \rightarrow \mathbb{N}$ , we say that  $M$  operates in time  $f(n)$ .

A language  $L$  belongs to the complexity class  $\text{TIME}(f(n))$  if  $L$  is decided by a multi-string TM operating in time  $f(n)$ . A time complexity class is the set of languages that can be decided within a certain time threshold.

A  $k$ -string Turing machine *with input and output* is a standard  $k$ -string Turing machine with the constraint that the input string is a read-only string, and the output string is a write-only string. Given a  $k$ -string Turing machine  $M$  with input and output, suppose that  $M$  halts in the configuration  $(s, \sigma_1, \sigma'_1, \dots, \sigma_k, \sigma'_k)$  on input  $x$ . The space required by  $M$  on input  $x$  is defined as  $\sum_{i=2}^{k-1} |\sigma_i \sigma'_i|$ , i.e. the space required is the sum of the lengths of all the strings excluding the input and the output string. A language  $L$  belongs to the complexity class  $\text{SPACE}(f(n))$  if  $L$  is decided by a  $k$ -string Turing machine with input and output operating in space  $f(n)$ .

A non-deterministic Turing machine is a tuple  $M = (S, \Sigma, \Delta, s)$ , where  $S, \Sigma$  and  $s$  are as in standard Turing machines, and  $\Delta$  is a transition relation:  $\Delta \subseteq (S \times \Sigma) \times [(S \cup \{\text{yes, no, h}\}) \times \Sigma \times \{\leftarrow, \rightarrow, -\}]$ . For each configuration, there may be more than one possible “next” configuration. Non-deterministic Turing machines differ from deterministic machines with respect to the definition of complexity classes. A non-deterministic Turing machine  $M$  is said to decide language  $L$  if, given  $x \in L$ ,  $M(x) = \text{yes}$  for a possible computation of  $M$ . Notice that  $M$  is not required to accept  $x$  in all possible computations. A non-deterministic Turing machine  $M$  decides a language  $L$  in time  $f(n)$  if (i)  $M$  decides  $L$  and (ii)  $M$  does not have computation paths longer than  $f(n)$  (where  $n$  is the size of the input). The set of languages decided by a non-deterministic Turing machine within time  $f(n)$  is denoted by  $\text{NTIME}(f(n))$ . The complexity class  $\text{NSPACE}(f(n))$  is defined analogously to  $\text{SPACE}(f(n))$ . Table 1.2 lists the definition of some commonly used complexity classes and their names.

Complexity classes can be ordered, as showed as follows:

$$\mathbf{L} \subseteq \mathbf{NL} \subseteq \mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE} \subseteq \mathbf{EXP}.$$

We know that  $\mathbf{L}$  is strictly a subset of  $\mathbf{PSPACE}$ , and that  $\mathbf{P}$  is strictly a subset of  $\mathbf{EXP}$ . At present, it is not known if the other inclusions are proper or not.

Name	Definition
<b>L</b>	$\bigcup \text{SPACE}(\log(n))$
<b>NL</b>	$\bigcup \text{NSPACE}(\log(n))$
<b>P</b>	$\bigcup \text{TIME}(n^k)$
<b>NP</b>	$\bigcup \text{NTIME}(n^k)$
<b>PSPACE</b>	$\bigcup \text{SPACE}(n^k)$
<b>NPSPACE</b>	$\bigcup \text{NSPACE}(n^k)$
<b>EXP</b>	$\bigcup \text{TIME}(2^{n^k})$

Table 1.2: Some complexity classes and their definitions.

### 1.3 Systems Engineering

Systems Engineering (SE) emerged as discipline in 1950s due to necessity of the U.S. Department of Defense of managing complex military equipments. This discipline is intended to help design and management of complex systems, namely systems with a high number of heterogeneous components that interact in many ways. Complex systems are characterized by the combinatorial explosive number of possible configurations, by the variety of different technologies by which are permeated and by the intense interaction with human users (that makes the human factor a key factor of SE). From that initial period, up to now, SE has spread over a wide range of fields of application (energy, transportation systems, spacecraft design, software integration . . .) and has showed a new approach to systems design and problem solving in general.

SE has an *holistic* approach. It adopts an high-level point of view in order to have a global vision of its target. SE focuses on a system as a whole. In particular, it does not observe a system apart from its environment. Secondly, SE uses a *top-down* methodology. In fact by considering the system as a whole, the main issues are addressed globally in the initial phases of the design process. Then, progressively breaking down the system into smaller components, systems engineers go down into details and identify interactions and interfaces between components.

In this work we adopt both these two methodological attitudes. Nevertheless, since SE is a very wide discipline, in the following sections we present only a few concepts that are necessary to our exposition (for a comprehensive introduction see [151, 15, 126, 131]).

#### 1.3.1 Requirements engineering

The International Council on Systems Engineering (INCOSE) defines systems as “*an interacting combination of elements organized to achieve one more stated purposes*”. Among several available definitions, this one stresses the importance of having stated an aim in order to define an (artificial) system<sup>3</sup>. We use the terms *system-of-interest*

<sup>3</sup>It fits only for designed systems, not natural ones, see Section 5 for a discussion.

or *target system* or *system-to-be*, equivalently, to refer to the system whose design is under consideration.

Rarely systems can be described once for all by means of a single representation. From a methodological point of view, real systems (especially complex ones) are described adopting different perspectives and different levels of detail. Normally, several models are necessary; one for each *point of view* and *level of abstraction*. The set of points of view that designers choose to describe the target system, defines their conceptual systems architecture framework. An architecture framework is an encompassing structure, or set of structures, which can be used for developing a set of different system architectures, since it includes methods, tools and standards for representing and designing systems. It provides the frame of reference for a consistent organization of the many models that can be required during the systems design process. “Classical” architecture frameworks are, among others, DODAF, MODAF, TOGAF, Zachman<sup>4</sup>. We adopt the systems architecture framework proposed in [16, 17, 95, 74].

Some techniques help to manage complex systems decoupling them into parts. In this context, we use the terms *abstraction* and *concretization* to indicate the processes that allow to shift from a specific level of details of the target system to a properly lower/higher one, so that we can focus on the features which are really relevant and “hide” the others. *Decomposition* and *integration* are the ones that let the analyst identify and solve sub-problems. Alternating these techniques, we can provide a series of partial *visions* that cooperate to produce the whole picture, instead of a single global description. We call *operational vision* what concerns the analysis of the external interactions of the system and its aim, *functional vision* what regards the analysis of the abstract functions of the system and *constructional vision* what deals with the analysis of the concrete components of the system. A fundamental distinction is between the *internal* and the *external* perspective of the system. When designers take an internal point of view, they focus on the capability of a system to perform correctly a given function. They take care of ensuring that the system works correctly. Functional and constructional visions are internal. When designers assume an external point of view, they try to empathize with the stakeholders and to identify the reasons why a system is needed. Operational vision is external.

The Institute of Electrical and Electronics Engineers (IEEE) proposes the following definition of the term *requirement*<sup>5</sup>.

**Definition 1.3.1 (Requirement)** *A requirement  $R(S)$  on a system  $S$  is a non ambiguous, testable and measurable feature which expresses an operational, functional or constructional characteristics of  $S$  or a constraint on  $S$  which is necessary for the approval of the system  $S$  by its stakeholders.*

<sup>4</sup>See <http://www.iso-architecture.org/ieee-1471/afs/frameworks-table.html> for an overview

<sup>5</sup>[IEEE Std 1220-1998]

System's requirements have to be distinguished from users' and stakeholders' needs and from specifications. In fact, the term *requirement* shall be used to indicate only something that is inside the system domain, while the term *need* shall be used to indicate a desideratum that is entered by a stakeholder. Requirements concern systems, needs concern stakeholders. In particular, system's requirements are answers to stakeholders' needs. Moreover, Nuseibeh et al. [130] specify that "*requirements are things in the application domain that we wish to be made true by delivering the proposed system*" while "*a specification is a description of the behaviors that the system must have in order to meet the requirements*".

The development of a system is always motivated by desiderata that are expressed by customers, end-users and other stakeholders who are external to the system to be realized. The first task of system designers is to identify and organize these desiderata and to specify them in a clear set of requirements.

Requirements have to be organized coherently, allocated to the proper level of analysis (since they can concern the whole system but also subsystems and components) and traceable (i.e. it shall be possible to trace the connections between requirements and needs, in order to explain which is the motivation of each component of a system). All these activities are included in the scope of a specific discipline, called *Requirements Engineering* (RE). The IEEE defines it as follows<sup>6</sup>.

**Definition 1.3.2 (Requirements Engineering)** (1) *The process of studying user needs to arrive at a definition of system, hardware, or software requirements.* (2) *The process of studying and refining system, hardware or software requirements.*

Notice that, the term "requirements engineering" is used to denote both the discipline and the process. RE includes the following subfields.

- Requirements elicitation: it is the set of activities concerned with the issue of collecting the requirements. It has the objective of understanding the stakeholders' needs. The identification of the system boundaries and of the set of stakeholders are preliminary tasks. Typically, customers and end-users can be found easily, but other stakeholders can be difficult to identify.
- Requirements analysis: it is the set of actions that are done in order to classify requirements in homogeneous groups and identify possible conflicts among requirements of different stakeholders.
- Requirement prioritization: this subfield deals with the problem of ranking the requirements and establishing which ones have to be considered first, at an intermediate phase of a project or in a specific product release.

---

<sup>6</sup>[IEEE-Standard - 610.12, 91]

- Requirements traceability: it is concerned with the issue of tracking the evolution of requirements and tracing the links between them. It makes possible, for stakeholders, to find the motivation of a requirement even when several changes have been made to a project or system.

There are multiple taxonomies for requirements. Consistently with the terminology introduced above, we use the term *operational requirements* to refer to stakeholders' needs, *functional requirements* to refer to the processes that have to be performed by the system and *constructional requirements* to indicate the required concrete features of the actual elements of the system.

A common distinction that we take from RE literature, is between behavioral and non-behavioral (or quality) requirements. Behavioral requirements describe what a system shall do, non-behavioral requirements qualify the way a system accomplishes its tasks and its quality<sup>7</sup>. Non-behavioral requirements are called, informally, “-ilities” or “-ities” because many of them correspond to features that are referred by words that end with “-ility” or “-ity”, as accessibility, exploitability, extensibility, operability portability, security, usability. Of course, this is not always the case, for example coherence is a possible non-behavioral requirement. However, “-ilities” is a common way to refer to the whole set of these requirements, in the RE community. Some authors propose constraining taxonomies for quality constraints (for example, Fig. 1.1, coming from [18], shows a possible hierarchy of quality requirements). However, this approach is not unanimously accepted and most authors are more encompassing and admit a huge set of non-behavioral requirements. Moreover, the distinction between behavioral and non-behavioral requirements is fuzzy. For example, access control issues are generally considered to be a security issue and are classified as non-behavioral, even if they imply several actions that the system shall do, and, consequently, entail behavioral requirements. Requirements classification should be simply considered as a conceptual tool that helps their elicitation. Thus, we concur for a supple use of the distinction between behavioral and non-behavioral requirements.

Requirements can have different levels of abstraction. During the system design process, requirements are progressively refined and associated to sub-systems and components. At the first step of this process, requirements concern the entire system. They are referred as system-level or high-level requirements. Then, consistently with the process, they are classified as subsystem-level and component- (or low-) level. High-level requirements are the most abstract, in the sense that they do not concern concrete details, that are embodied by low-level requirements.

Requirements can be expressed in several, different ways. Possible approaches range from the utilization of natural language, to the use of semi-formal languages as

---

<sup>7</sup>Some authors use the term “non-functional requirement” as synonym of “non-behavioral requirement”, but we avoid this terminology that we consider misleading. The term “non-functional” suggests the idea that there is something that doesn't work.

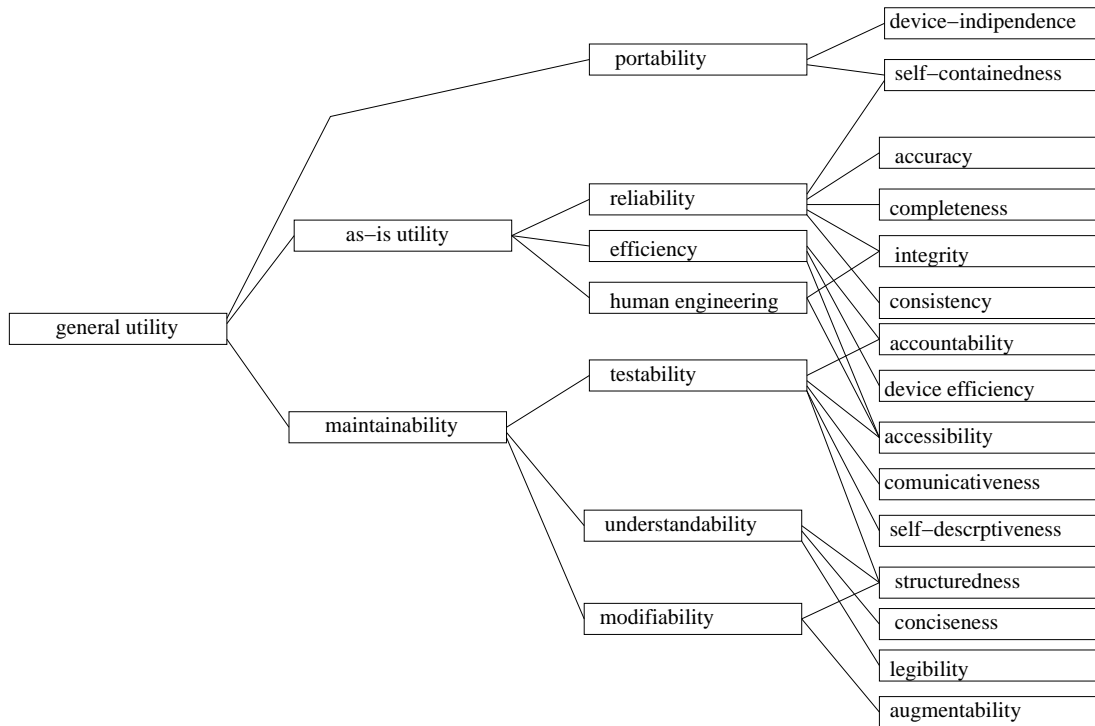


Figure 1.1: Quality requirements [18]

UML and SYSML, to formal methods and propositional logic. Nevertheless, in the early phases of requirements elicitation, they are usually written in natural language, respecting typical general patterns.

For operational requirements (i.e. needs), the generic informal pattern is:

- The external system  $S_n$  shall do  $f$

where  $f$  expresses a function of the external system  $S_n$ , i.e. a system in the environment that interacts with the target system and that requires a contribution (generally speaking, an answer) from the system-to-be.

For behavioral (functional) requirements, the generic informal pattern is:

- The target system  $S$  shall do  $f$

where  $f$  expresses a required function of the system.

For quality requirements, the generic informal pattern is:

- The target system  $S$  shall be  $Q$

where  $Q$  is a quality feature (an “-ilities”, in the sense introduced above).

For constructional requirements, the generic informal pattern is:

- The target system  $S$  shall be  $C$

where  $C$  is a constructional feature (a detailed description of a concrete technical characteristic as, for example “made in plastic” or “written in Java”).



These formulations are acceptable, but basic. They can be refined specifying context and performance level, in particular for behavioral requirements. The context defines the conditions in which a function is required. For example, if we think of a train, it shall prevent passengers to exit the wagon, when the train is moving, but the system shall facilitate passengers to leave when the train is at the station (where “moving” and “at the station” are the contexts). Typical contexts are, for example, “nominal”, “standby” and “maintenance”.

When the concerned functionality can be measured, the performance level assesses the threshold that has to be achieved in order to fulfil the requirement. For example, a train can be required to transport at least  $n$  passengers per hour from station A to station B. Context and performance level are meaningful, but, in reality are often omitted in practice.

Thus, for behavioral requirements, the full, generic informal pattern is:

- The system  $S$  shall do  $f$  in the context  $C$  with the performance  $P$

This pattern has a wide acceptance. Nevertheless, we notice that it embodies an approach to needs that can be termed “*satisficing*”<sup>8</sup> (a concept introduced by H.A. Simon, in his work [162, 163]) since it asks that a certain function is provided with a sufficient level of performance. Thus, a possible alternative way, to formulate the behavioral requirements of a system in optimization oriented way, is as follows:

- The system  $S$  shall do  $f$  in the context  $C$  with the highest possible performance

This formulation fits with an optimization oriented approach, which searches for optimal solutions. We remark that this formulation is not standard and not used in common SE practice. Nevertheless, since we integrate high-level requirements in MP formulations we adopt, implicitly, this kind of approach.

### 1.3.2 Goals

A goal is a task that the system should achieve, and represents the reason why an action is needed. Needs (i.e. operational requirements) are goals from the point of view of stakeholders and external systems. Functional requirements are goals from the point of view of the systems designer. In the following we use the concept of “goal” as a super-class that includes both operational and functional requirements (when the context is sufficiently clear to avoid confusion). Table 1.3 resumes our terminology and represents our framework. Each cell refers to a kind of requirement. Columns progress from *external* (the environment of the system) to *internal* (the system itself and its elements), rows from *abstract* (system-level) to *concrete* (component-level). In this work we focus on the the top-left cells of this schema, which correspond to system-level operational and functional requirements, i.e. global goals. Empty cells denote the type of requirements that we do not consider.

<sup>8</sup>Satisfice means satisfy and suffice.



	Needs (operational requirements)	Functional requirements	Constructional requirements
	external	internal	internal
system-level	global goals	global goals	
sub-system-level			
...			
component-level			

Table 1.3: Global Goals: high-level needs and high-level functional requirements

Goal-Oriented Requirements Engineering (GORE), is a sub-field of RE that focuses on goals analysis. GORE analyzes a target system focusing on the aims, expectations and needs in order to define properly a set goals. There are several GORE methods. The following list resumes the principal ones.

- The NFR<sup>9</sup> Framework focuses on the classification of non-behavioral requirements. It provides a “map” that entails a special care of non-behavioral requirements during the initial phases of systems design [38,125].
- $i^*$  is a modeling framework that can be used to represent the reasons that motivate the realization of a system. In particular, it is used to represent the environment of a system, its stakeholders and their objectives [177].
- The KAOS methodology (Knowledge Acquisition in autOdated Specification) is a framework that mixes semi-formal representation tools and formal ones. It is based on semantic nets and linear-time temporal logic [45].
- The Tropos methodology [71,29] (that we present in Section 1.3.2.1, since we borrow from it some concepts and tools used in this work).

### 1.3.2.1 Tropos

*Tropos*<sup>10</sup> is a semi-formal language and modeling methodology inspired by the  $i^*$  methodology. It endorses an agent oriented programming approach, and it exploits mentalistic notions as goals, beliefs and plans. The aim of Tropos is to ease the understanding of the environment and of the interactions among stakeholders of a system-to-be. It is intended to cover all the phases of requirements engineering (it has a wider range of applicability than  $i^*$ , see Fig. 1.2 that is from [124]).

The methodology covers five phases: early requirements management, late requirements management, architectural design, detailed design and implementation. However, if compared with other methods, for example UML-oriented ones, Tropos focuses more on the early phases of the system design process.

<sup>9</sup>NFR: Non-Functional Requirement

<sup>10</sup>From the Greek “*tropé*”, i.e. easily “changeable and adaptable”.

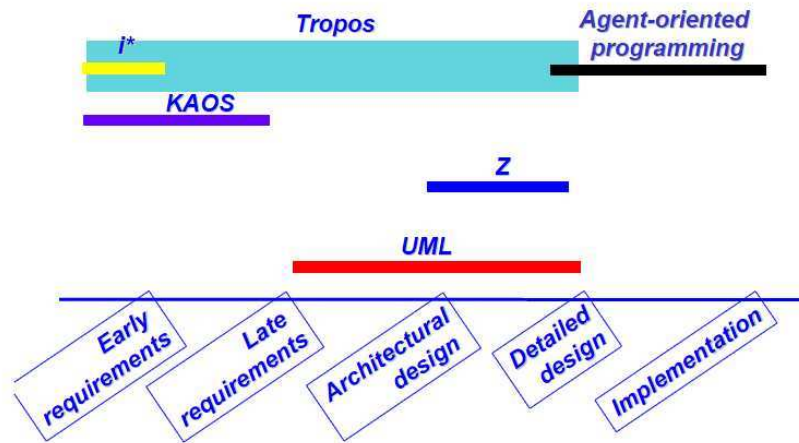


Figure 1.2: Tropos in perspective [124]

In order to provide an early requirements model, Tropos includes several modeling activities. Among them, the most relevant from our point of view, are the following ones. Notice that the term “actor”, in Tropos methodology context, means “an entity that has a goal”. Thus, the stakeholders of a system are actors.

- Actor modeling: this activity aims to identify actors of the environment and of the system.
- Dependency modeling: it aims to find which actors depends on another one for achieving a goal.
- Goal modeling: it aims to analyze goals to understand if they can be decomposed in subgoals and if there are dependencies among them.

In Tropos methodology, goals are classified as “hard” if they are sharply defined and there is a clear condition that states if the goal is achieved or not. Otherwise, goals are termed “soft”.

During goal modeling, a goal can be broken down into subgoals that cover it. This means that subgoals entail the main goal. If all subgoals have to be fulfilled to achieve the original one, then the decomposition is called AND-decomposition. If the fulfillment of at least one subgoal entails the achievement of the main goal, the decomposition is called OR-decomposition. However, when relationships among goals are not clearly defined, these decompositions can be not suitable. This happens more frequently with soft-goals. Moreover AND/OR decomposition does not allow the expression of the opposition of two goals. For this reason the methodology includes other relationships. In particular, it admits the *positive contribution*, labeled by the “+” symbol, and the *negative contribution*, labeled by the “-” symbol. Given two goals  $G_1$  and  $G_2$ ,  $+(G_1, G_2)$  means that the achievement of  $G_1$  helps the achievement of  $G_2$ . Conversely,  $-(G_1, G_2)$  means that the achievement of  $G_1$  con-

trasts the achievement of  $G_2$ . In this case, as we discuss in the following, a trade-off between  $G_1$  and  $G_2$  has to be managed.

Actor and goal models are represented graphically by means of actor and goal diagrams. Actors are depicted as circles, hard goals as ovals, softgoal as cloud shapes. Arrows links show the relationships among actors and goals (see Fig. 1.21). We use goal diagrams of Tropos methodology to represent in semi-formal way the objectives of the problems we deal with in this work.

### 1.3.3 Systems design and management process

Complex systems design and management are processes that can last very long. The concerned period of time can be split in typical phases and there are several frameworks that propose possible subdivisions. A first, big division is between the acquisition phase and the in-service phase. In the acquisition phase the system is designed and built; in the in-service phase, the system works, is managed and maintained. Fig. 1.3 (that we borrow from [15]) shows a general process.

Widespread design processes are, among others, the *waterfall*, *spiral* and *V-Model* models. Originally conceived for information systems, are now used in several industrial fields. The *waterfall model*, showed in Fig. 1.4, is the earliest method

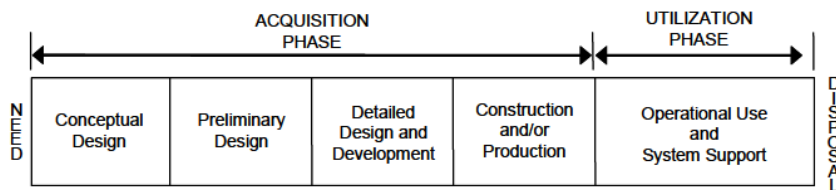


Figure 1.3: System life cycle.

for system development process management. It consists in subsequent steps that progress in rigid sequence. This means that a phase can begin only when the previous is completely ended and there is not possible rollback (the name waterfall, suggests exactly this idea).

The *spiral model*, depicted in Fig. 1.5, is based on a cyclic repetition of all the development phases with an increasing level of detail and concreteness. All phases are considered once in order to produce a first prototype or model, and then re-executed  $n$  times, producing  $n$  prototypes or models, progressively more accurate. It allows for incremental improvement at each “turn around the spiral”.

The *V-model*, showed in Fig. 1.6, is used to underline the relationships between development and testing phases. The descending-ascending disposition of the phases of the development cycle in its diagram gives it its particular V shape. The descending part of the “V” includes requirements management and design, and the ascending part includes integration, verification and validation. The symmetry of the dispo-

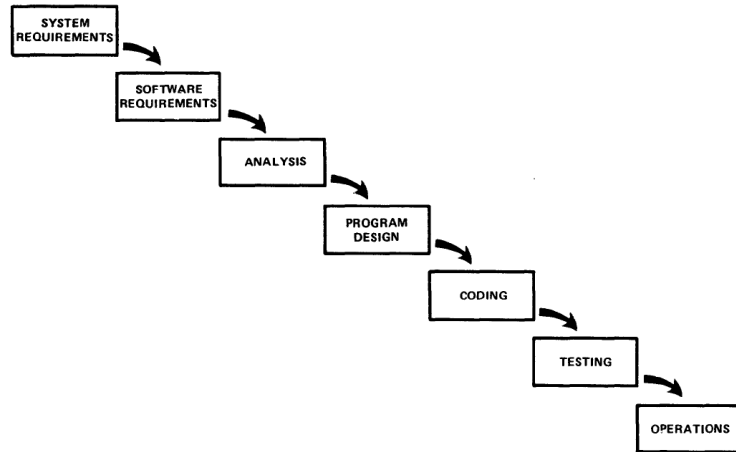


Figure 1.4: Waterfall model, from Royce (1970), see [150]

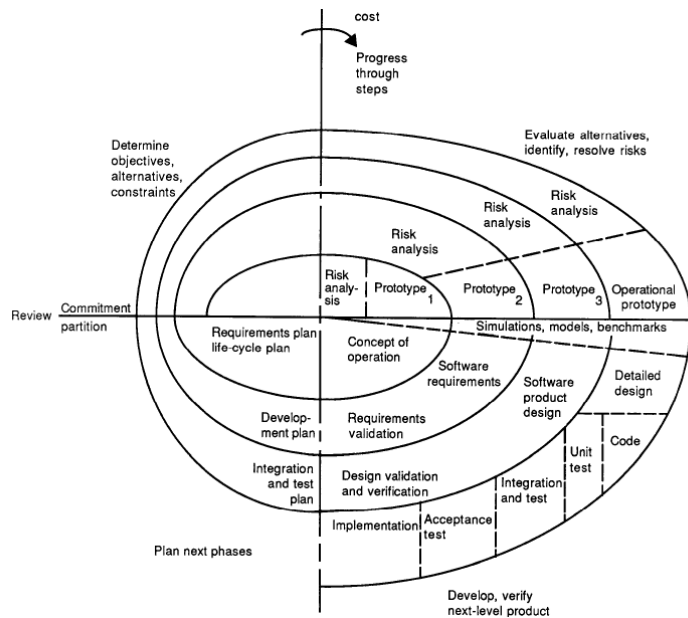


Figure 1.5: Spiral model, from Boehm (1988), see [19]

sition suggests that there should be a continuous interchange among early and late steps of the design.

The *Manifesto for Agile Software Development* [11] introduces in the context of software systems development the concept of *agile methods*. This kind of approach is gaining increasing consideration in the last years and it is what we might call an “hot topic”. The agile methodology appears to have several positive features that are useful for many types of systems (principally, but not only, software ones). It is an iterative approach that was proposed to avoid the shortcomings of waterfall methods and one of its feature is the tightening of the design-develop-test loops. Ideas already proposed by spiral and V models become more radical. Development sub-cycles are short and many, frequent prototypes are tested in order to manage

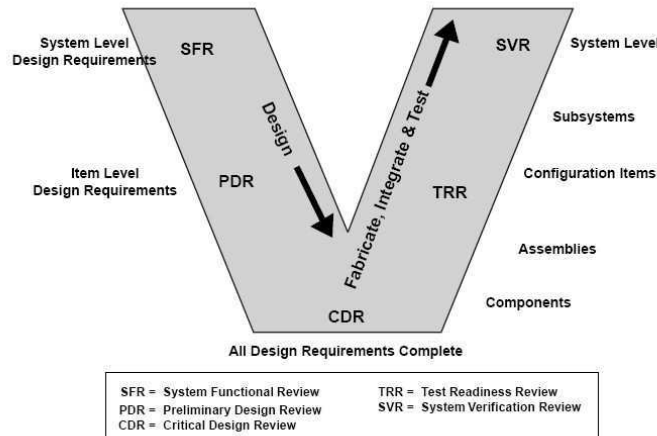


Figure 1.6: V-model, see [8]

the correctness of the design process [1].

### 1.3.4 From goals to objective functions

One of the hypothesis of our work is the adoption of an iterative system design process, represented, for example by the *spiral-model* introduced in Section 1.3.3. We figure out a scenario in which we want to design a system with this kind of approach. This includes the production of several prototypes, with different abstraction level; at least, one for each iteration. In particular we focus on the first “turn”, which aims to produce the first prototype of the system-to-be, considering only high-level requirements.

As mentioned in the previous sections, often, high-level requirements, are stated as soft ones. Moreover, agreements and oppositions among goals are assumed on a hypothetical basis. Designers suppose that some goals could contrast some others. Nevertheless, it is only during the following steps, when formal models are deployed, that these assumptions can be verified. Thus, new trade-offs can be necessary and supposed ones can reveal themselves as apparent. In others words, potential trade-offs are commonly analyzed in details only when the design process reaches the constructional level.

	Hard	Soft
Do		Req. 1
Be		Req. 2

Figure 1.7: Basic requirements framework (a).

	Hard	Soft
Do	<b>Req. 1</b>	
Be	<b>Req. 2</b>	

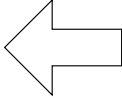


Figure 1.8: Basic requirements framework (b).

	Hard	Soft
Do	<b>Req. 1</b> <b>Req. 2 (ex)</b>	
Be		

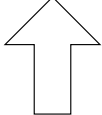


Figure 1.9: Basic requirements framework (c).

If, in the initial phases of the design process, a requirement is specified in a soft manner, then, it should be transformed in a hard one, i.e. one for which there is a measurable satisfaction criterion (see Fig. 1.7 and Fig. 1.8). Subsequently, if possible, requirements should be reformulated in behavioral terms (see Fig. 1.9). This can be done considering the fact that non-behavioral requirements of the target system, correspond to behavioral ones of an external system that is in its environment. This ideal path, i.e. reduction of requirements to hard, behavioral ones, guides our modeling process, as we will see in the next chapters. In fact, we integrate high-level requirements into MP formulations. These make them well defined, formal and, *ipso facto*, hard (measurable). As a (positive) consequence, trade-offs can be precisely analyzed early.

## 1.4 Structure of the work

This work considers the integration of (quality) high-level requirements in system optimization problems in order to provide a fast prototyping and to facilitate the design process. Its structure is as follows:

1. In Chapter 1 we provide motivational introduction and common background.

2. In Chapter 2, we present and formalize a problem arising in the strategic innovation management of service-based enterprises. The informal, system-level requirement we deal with is:

- The information system shall ensure profitable services and it shall be maintainable and extensible.

where *provide services* and *be profitable, maintainable and extensible* are respectively the behavioral and non-behavioral component of the system-level requirement. We provide a MP formulation of this problem, i.e. the evolution of an information system architecture. Model validation is approached with a collaborative methodology (see Chapter 5 for a discussion on this point). We ask a team of experts to judge their compliance with the empirical reality perceived by each of them, to decide if the model is valid.

3. In Chapter 3 we consider the problem of analyzing and improving *recommender systems*. The informal, system-level requirement we deal with is:

- The recommender system shall provide recommendations to users, about items of a given domain, and it shall be good, interesting and accurate.

where *provide recommendations* and *good, interesting and accurate* are respectively the behavioral and non-behavioral component of the requirement. We propose several models and adopt an experimental approach to select the best, choosing a benchmark to compare their performances. A model is *best* if the corresponding solutions are the best in the benchmark. This approach is empirical and requires the possibility of repeatable tests (see Chapter 5, for a discussion on this point).

4. In Chapter 4, we describe and solve a problem arising in the transportation of hazardous material on a road network, using a fleet of lorries. The informal, system-level requirement we deal with is:

- The transportation system shall ensure safe disposal of hazardous waste in such a way that the risk of potential catastrophic accident is equitable over the population.

where *ensure hazardous waste disposal* and *be equitable* are respectively the behavioral and non-behavioral component of the system-level requirement. We propose two formalizations of the concepts of equity and provide a rationale to choose between them.

5. In Chapter 5 we provide some epistemological and historical considerations related to the topics proposed in this thesis.

6. In Chapter 6 we present the conclusions of this work.


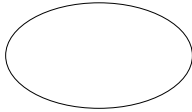
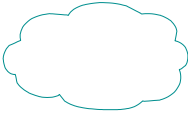

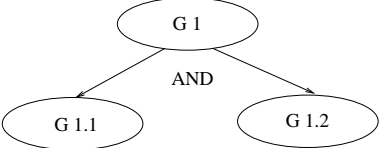
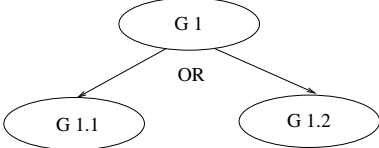
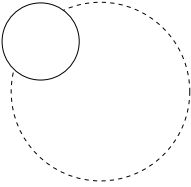
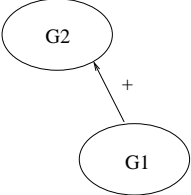
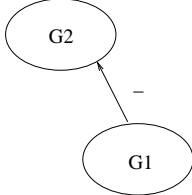
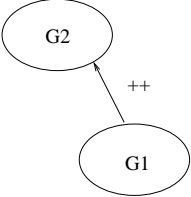
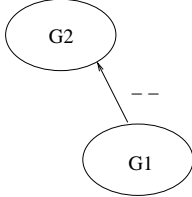
 <p>Figure 1.10: Actor</p>	 <p>Figure 1.11: Hard goal</p>
 <p>Figure 1.12: Soft goal</p>	 <p>Figure 1.13: Dependency: actor A 1 depends on actor A 2 to achieve the goal G</p>
 <p>Figure 1.14: AND-decomposition: goal G 1 derives into goal G 1.1 AND goal G 1.2</p>	 <p>Figure 1.15: OR-decomposition: goal G 1 derives into goal G 1.1 OR goal G 1.2</p>
 <p>Figure 1.16: Actor scope</p>	
 <p>Figure 1.17: Support</p>	 <p>Figure 1.18: Opposition</p>
 <p>Figure 1.19: Implication</p>	 <p>Figure 1.20: Break</p>

Figure 1.21: Tropos diagram components





**Part II**

**Applications**



# The information system architecture evolution problem

## 2.1 Introduction

For any information system manager, a recurrent key challenge is to avoid creating more complexity within its existing information system through the numerous IT projects that are launched in order to respond to the needs of the business. Such an objective leads thus typically to the necessity of co-optimizing both creation *and* replacement/destruction — called usually *kills* in the IT language — of parts of the information system, and of prioritizing the IT responses to the business consequently.

This important question is well known in practice and quite often addressed in the IT literature, but basically only from an enterprise architecture or an IT technical management perspective [14, 37, 115]. Architectural and managerial techniques, however, are often only parts of the puzzle that one has to solve to handle these optimization problems. On the basis of budget, resource and time constraints given by the enterprise management, architecture provides the business and IT structure of these problems. This is however not sufficient to model them completely or solve them. We propose an operational model and a Mathematical Programming (MP) formulation expressing a generic global prioritization problem occurring in the (limited, but practically relevant) context of a technological evolution of an information system.

The presence of many teams at work is worth a close examination, because it can lead to puzzling situations if some specific conditions hold or resources are insufficient. If we introduce a bound on the duration of the whole transition, or, at least, on each step of it, and force the activation/deactivations to be done before a short deadline we generate a “competition” between departments. Thus, we employ multi-optimization techniques in order to model and numerically solve a wider part of this general problem. This is a step towards a full formalization of the problem

which promises to provide a valuable help for IT practitioners.

The rest of this chapter is organized as follows: section 2 describes the problem and the key elements involved, section 3 proposes the Mathematical Programming formulation and introduces the necessary multi-optimization techniques, section 4 discusses the theoretical properties of the model and section 5 reports the results of the computational tests.

## 2.2 Operational model of an evolving information system

### 2.2.1 Elements of information system architecture

Any information system of an enterprise (consisting of a set  $D$  of departments) is classically described by two architectural layers:

- the *business layer*: the description of the business services (forming a set  $V$ ) offered by the information system;
- the *IT layer*: the description of the IT modules (forming a set  $U$ ) on which business services rely on.

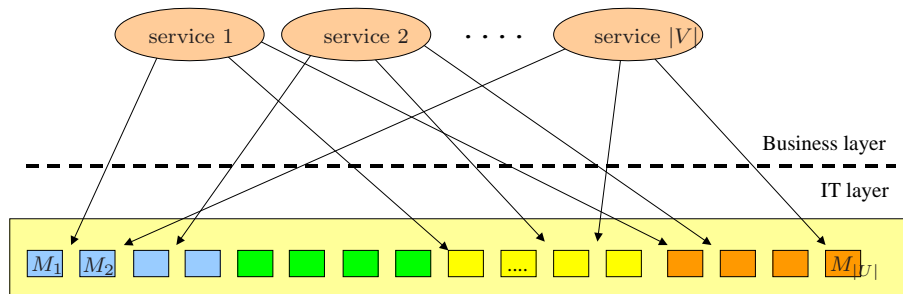


Figure 2.1: A simple two-layer information system architecture.

In general, the relationship  $A \subseteq V \times U$  between these two layers is not one-to-one. A given business service can require any number of IT modules to be delivered and vice-versa a given IT module can be involved in the delivery of several business services, as shown in Fig. 2.1.

### 2.2.2 Global Goal

The general goal, we deal with, is:

- The information system shall ensure profitable services and it shall be maintainable and extensible.

Fig. 2.2 shows the related goal diagram. *Provide services* is a behavioral requirement. The non-behavioral requirements that we consider are: *be profitable*, *be maintainable* and *be extensible*. *Be profitable* is a hard goal. *Be maintainable* and *be extensible* are soft-goals, which tell us that the target system shall be able to evolve. We need some more elements in order to make these soft-goals precise. We introduce them in the next sections.

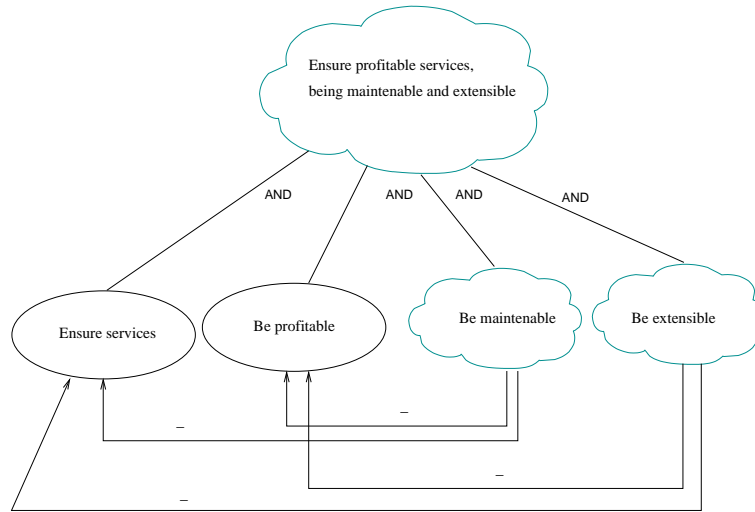


Figure 2.2: Goal Diagram

### 2.2.3 Evolution of an information system architecture

From time to time, an information system may evolve in its entirety due to the replacement of an existing software technology by a new one (e.g. passing from several independent/legacy software packages to an integrated one, migrating from an existing IT technology to a new one, and so on). These evolutions (or *transitions*) invariably have a strong impact at the IT layer level, where the existing IT modules  $U^E = \{M_1, \dots, M_n\}$  are replaced by new ones in a set  $U^N = \{N_1, \dots, N_{n'}\}$  (in the sequel, we assume  $U = U^E \cup U^N$ ). This translates to a replacement of existing services (sometimes denoted as ES) in  $V$  by new services (sometimes denoted as NS) in  $W$  ensuring that the impact on the whole enterprise is kept low in order to avoid business discontinuity. This also induces a relation  $B \subseteq W \times U^N$  expressing reliance of new services on IT modules. Note also that in this context, at the business level, there exists a relation (in  $V \times W$ ) between existing services and new services which expresses the fact that a given existing service shall be replaced by a subset of new business services. We note in passing that this relation also induces another relation in  $U^E \times U^N$  expressing the business covering of an existing IT module to a subset of new IT modules (see Fig. 2.3).

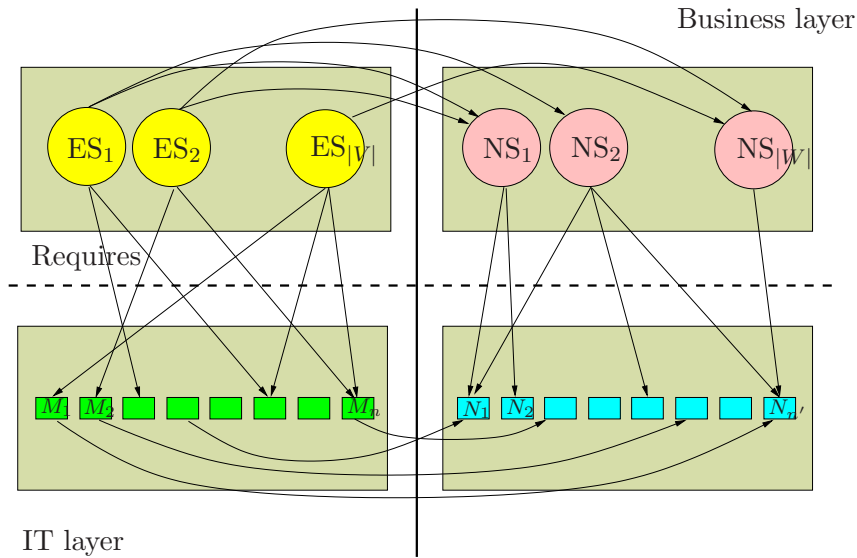


Figure 2.3: Evolution of an information system architecture.

#### 2.2.4 Management of IT system architecture evolutions

Mapping the above information system architecture on the organization of a company, it appears clear that three main types of enterprise actors are naturally involved in the management of these technological evolutions which are described below.

1. *Business department managers*: they are responsible of creating business value — within the perimeter of a business department in the set  $D$  — through the new business services. This value might be measured by the amount of money they are ready to invest in the creation of these services (business services are usually bought internally by their users within the enterprise).
2. *IT project managers*: they are responsible for creating the new IT modules, which is a pre-requisite to creating the associated business services. The IT project manager has a project schedule usually organized in work packages, each having specific starting times and global budget: in our case, this schedule is presented as a set of deployment precedence rules for new modules.
3. *Kill managers*: they are responsible for destroying the old IT modules in order to avoid to duplicate the information system — and therefore its operating costs — when achieving its evolution. Kill managers have a budget for realizing such “kills”, but they must ensure that any old IT module  $i$  is only killed after the new services replacing those old ones relying on  $i$  are put into service. The enterprise motivates the efficiency of kill managers by setting a monetary value on each deactivation: this provides a measure of the desirability of killing module  $i$ .

In this context, managing the technological evolution of an information system means being able of creating new IT modules within the time and budget constraints of the IT project manager in order to maximize both the IT modules business value brought by the new services and the associated kill value (i.e. the number of old services than can be killed).

### 2.2.5 The IT system architecture evolution management

The architecture evolution of the IT system involves revenues, costs and schedules over a time horizon  $t_{\max}$ , as detailed below.

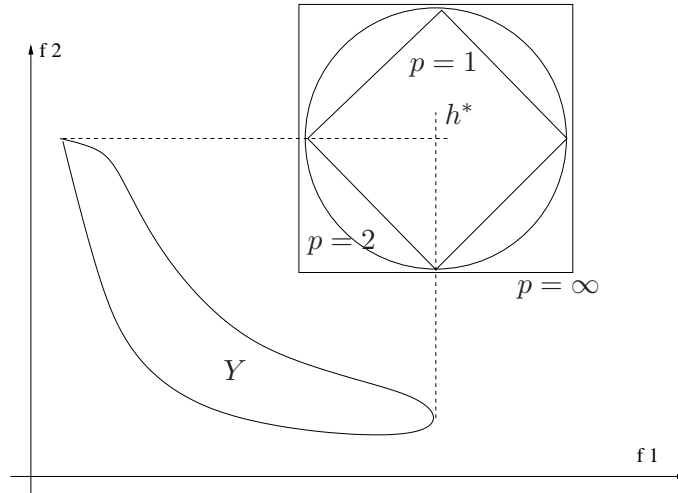
- *Time and budget constraints of the IT project manager.* Each new IT module  $i \in U$  has a cost  $a_i$  and a production schedule.
- *IT module business value.* Each department  $\ell \in D$  is willing to pay  $q_{\ell k}$  monetary units for a new service  $k \in W$  from a departmental production budget  $H^\ell = \sum_{k:(\ell,k) \in F} q_{\ell k}$ ; the business value of the new service  $k$  is  $c_k = \sum_{\ell:(\ell,k) \in F} q_{\ell k}$ . We assume that this business value is transferred in a conservative way via the relation  $B$  to the IT modules. Thus, there is a business contribution  $\beta_{ik}$  over every  $(i, k) \in B$  such that for each  $k$  we have  $c_k = \sum_{(i,k) \in B} \beta_{ik}$ ; furthermore, the global business value of module  $i$  is  $\sum_{k:(i,k) \in B} \beta_{ik}$ . We also introduce a set  $N \subseteq U$  of IT modules that are necessary to the new services.
- *Deployment schedule of new modules.* We are given a Directed Acyclic Graph (DAG)  $(U, S)$  where each couple  $(i, h) \in S \subset U \times U$  encodes a deployment precedence between the new modules  $i, h$  (i.e.  $h$  cannot be deployed before  $i$ ).
- *Kill value.* Discontinuing (or *killing*) a module  $i \in U$  has a cost  $b_i$  due to the requirement, prior to the kill, of an analysis of the interactions between the module and the rest of the system architecture, in order to minimize the chances of the kill causing unexpected system behaviour. As mentioned above, it also has a monetary value (or desirability)  $\phi_i$ .

The evolution involves several stakeholders. The department heads want to maximize the value of the required new services. The module managers want to produce the modules according to an assigned schedule whilst maximizing the business value for the new services to be activated. The kill managers want to maximize the monetary value of the deactivated modules within a certain kill budget. Thus, the rational planning of this evolution requires the solution of an optimization problem with several constraints and criteria, which we shall discuss in the next session.

## 2.3 Mathematical Programming based approach

We present two MOP techniques that we use in the following. The  $L_p$ -metric method sketched below belongs the class of *no-preference* methods. It aims to identify a



Figure 2.4:  $L_p$ -metric method

point in the objective space which has minimal distance from a reference point  $h^*$ . Typically, the chosen reference is the *ideal point (utopia)*, which corresponds to the maximum values of all objectives, optimized separately. In other words, the ideal point is the one we would choose if there were no compromise between tasks. For a vector  $f = (f_1, \dots, f_k)$  of objective functions, the ideal point is formally defined as the  $h^* \in \mathbb{R}^k$  satisfying:

$$\forall i \in \{1, \dots, k\} \quad h_i^* = \max\{f_i(s) \mid s \in X\}, \quad (2.1)$$

where depending on the decision variables  $s$  and  $X$  is the feasible region of the MOP. Figure 2.4 shows the sets of points that are equidistant from  $h^*$ , determined by different norms. The circle is given by the common Euclidean norm. The square and the rhombus are determined by the Maximum norm ( $p = \infty$ ) and the Manhattan norm ( $p = 1$ ). Thus, if we use the utopia point as benchmark and the  $p$ -norm to define a metric, the general formulation of the method is:

$$\min \left( \sum_{i=1}^k |f_i(x) - h_i^*|^p \right)^{1/p} \quad (2.2)$$

We remark that if we consider  $p = 1$ , we obtain the single objective problem:  $\min_s \sum_i |h_i^* - f_i(s)| = \min_s \sum_i (h_i^* - f_i(s)) = \|h^*\|_1 + \max_s \sum_i f_i(s)$ , since  $\|h^*\|_1$  is a constant. It therefore suffices to maximize  $\sum_i f_i(s)$  with respect to  $s$ .

The weighted-sum method is (probably) the most common “*a posteriori*” approach to multi-objective optimization. Its principle is to assign to each individual objective function a weight  $\alpha_i \geq 0$  normalized by  $\sum_{i=1}^k \alpha_i = 1$  and then to replace the set of objective functions by the single compound objective  $\min_s \sum_{i=1}^k \alpha_i f_i(s)$ . By varying  $\alpha = (\alpha_1, \alpha_2, \dots, \alpha_k)^T$  one can obtain a subset of the total set of efficient solutions (see Figure 2.5 and 2.6). If all of the weights are positive, the minimum of

the scalar-valued formulation is an efficient solution. Conversely, there may be efficient solutions which cannot be found by minimizing any weight vector  $\alpha$  (this might happen whenever the feasible set is nonconvex). In such cases, the weighted-sum method can be used to find an approximation of the efficient frontier. If necessary, other techniques can be applied in the second phase to inspect the objective space between solutions found by the weighted-sum method.

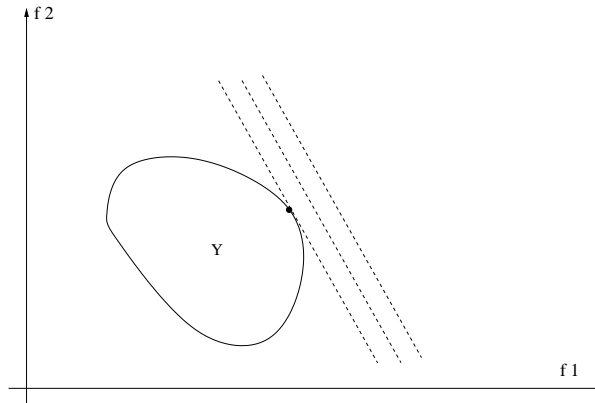


Figure 2.5: Weighted-sum method (1)

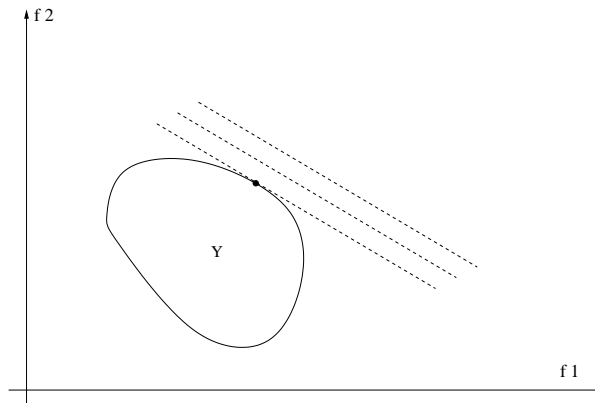


Figure 2.6: Weighted-sum method (2)

### 2.3.1 Introducing the MP formulation

As explained above, an enterprise in our context consists of a set  $D$  of departments currently relying on existing services in  $V$  and wishing to evolve to new services in  $W$  within a time horizon  $t_{\max}$ . Each service relies on some IT module in  $U$ . The set  $N \subseteq U$  indexes those IT modules that are necessary. The relations between services and modules and, respectively, departments and services, are denoted as follows:  $A \subseteq V \times U$ ,  $B \subseteq W \times U$ ,  $E \subseteq D \times V$  and  $F \subseteq D \times W$ . If an IT module  $i \in U$  is required by a new service, then it must be produced (or activated) at a certain cost  $a_i$ . When an IT module  $i \in U$  is no longer used by any service it must be killed at a certain cost  $b_i$ . Departments can discontinue using their existing

services only when all new services providing the functionalities have been activated; when this happens, the service (and the corresponding IT modules) can be killed; a killed module  $i$  contributes  $\phi_i$  monetary units to the goal of the kill manager. Departments have budgets dedicated to producing and killing IT modules, which must be sufficient to perform their evolution to the new services; for the purposes of this paper, we suppose that departmental budgets are interchangeable, i.e. all departments credit and debit their costs and revenues to two unique enterprise-level budgets: a production budget  $H_t$  and a kill budget  $K_t$  indexed by the time period  $t$ . A new service  $k \in W$  has a value  $c_k$ , and an IT module  $i \in U$  contributes  $\beta_{ik}$  to the value of the new service  $k$  that relies on it. We use the graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  shown in Fig. 2.7 to model departments, existing services, new services, IT modules and their relations. The vertices are  $\mathcal{V} = U \cup V \cup W \cup D$ , and the edges are  $\mathcal{E} = A \cup B \cup E \cup F$ . This graph is the union of the four bipartite graphs  $(U, V, A)$ ,  $(U, W, B)$ ,  $(D, V, E)$  and  $(D, W, F)$  encoding the respective relations. We remark that  $E$  and  $F$  collectively induce a relation between existing services and new services with a “replacement” semantics (an existing service can be killed if the related new services are active).

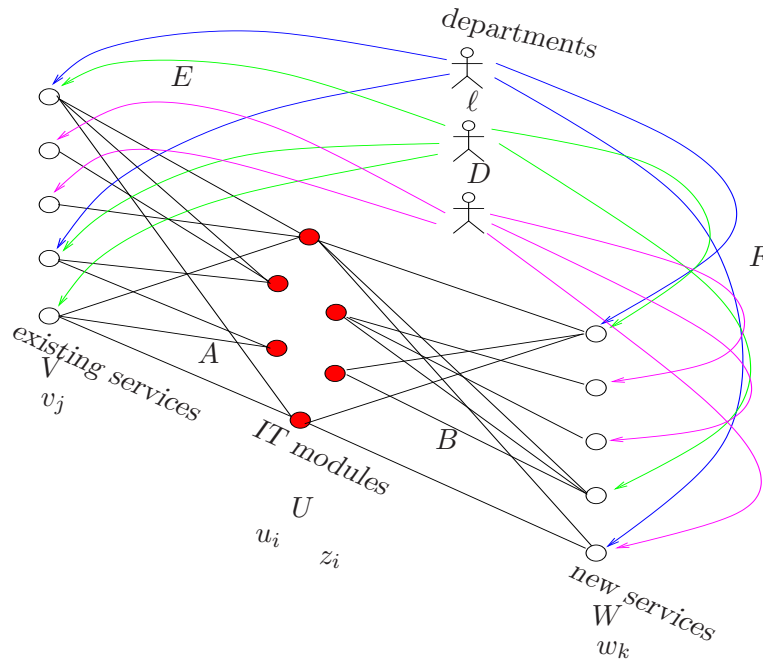


Figure 2.7: The bipartite graphs used to model the problem.

### 2.3.2 Sets, variables, objectives, constraints

We present here the MP formulation of the Architecture Evolution Problem (AEP). We recall that NS stands for new service and ES for existing service.

#### 1. Sets:

- $T = \{0, \dots, t_{\max}\}$ : set of time periods (Sect. 2.2.5, p. 35);
- $U$ : set of IT modules (Sect. 2.2.1, p. 32);
- $N \subseteq U$ : set of IT modules that are necessary for the NS (Sect. 2.2.5, p. 35);
- $V$ : set of existing services (Sect. 2.2.1, p. 32);
- $W$ : set of new services (Sect. 2.2.3, p. 33);
- $A \subseteq V \times U$ : relations between ES and IT modules (Sect. 2.2.1, p. 32);
- $B \subseteq W \times U$ : relations between NS and IT modules (Sect. 2.2.3, p. 33);
- $D$ : set of departments (Sect. 2.2.1, p. 32);
- $E \subseteq D \times V$ : relations between departments and ES (Sect. 2.3.1, p. 38);
- $F \subseteq D \times W$ : relations between departments and NS (Sect. 2.3.1, p. 38);
- $S \subset N \times N$ : deployment precedences between new modules (Sect. 2.2.5, p. 35).

## 2. Parameters:

- $\forall i \in U a_i =$  cost of producing an IT module (Sect. 2.2.5, p. 35);
- $\forall i \in U b_i =$  cost of killing an IT module (Sect. 2.2.5, p. 35);
- $\forall i \in U \phi_i =$  desirability (monetary units) of killing an IT module (Sect 2.2.5, p. 35)
- $\forall t \in T H_t =$  production budget per time period (Sect. 2.3.1, p. 38);
- $\forall t \in T K_t =$  kill budget per time period (Sect. 2.3.1, p. 38);
- $\forall (i, k) \in B \beta_{ik} =$  monetary value given to NS  $k$  by IT module  $i$  (Sect. 35, p. 2.2.5).

## 3. Decision variables:

$$\forall i \in U, t \in T u_{it} = \begin{cases} 1 & \text{if IT module } i \text{ is used for a ES at time } t \\ 0 & \text{otherwise;} \end{cases} \quad (2.3)$$

$$\forall i \in U, t \in T z_{it} = \begin{cases} 1 & \text{if IT module } i \text{ is used for a NS at time } t \\ 0 & \text{otherwise;} \end{cases} \quad (2.4)$$

$$\forall j \in V, t \in T v_{jt} = \begin{cases} 1 & \text{if existing service } j \text{ is active at time } t \\ 0 & \text{otherwise;} \end{cases} \quad (2.5)$$

$$\forall k \in W, t \in T w_{kt} = \begin{cases} 1 & \text{if new service } k \text{ is active at time } t \\ 0 & \text{otherwise.} \end{cases} \quad (2.6)$$

## 4. Objective functions.

- Business gain: value contributed to new services by IT modules. This is the objective of the module managers, and, because  $\beta$  is indexed on both modules and services, it agrees with the objective of the department heads.

$$\max_{u,v,w,z} \sum_{\substack{t \in T \\ (i,k) \in B}} \beta_{ik} z_{it} w_{kt}. \quad (2.7)$$

- Killing gain: objective of the kill managers.

$$\max_{u,v,w,z} \sum_{\substack{t \in T \\ i \in U}} \phi_i (1 - u_{it}). \quad (2.8)$$

## 5. Constraints.

- Production budget (cost of producing new IT modules; this is another objective of the module managers):

$$\forall t \in T \setminus \{t_{\max}\} \quad \sum_{i \in U} a_i (z_{i,t+1} - z_{it}) \leq H_t, \quad (2.9)$$

where the term  $z_{i,t+1} - z_{it}$  is only ever 1 when a new service requires production of an IT module — we remark that the next constraints prevent the term from ever taking value  $-1$ .

- Once an IT module is activated, do not deactivate it.

$$\forall t \in T \setminus \{t_{\max}\}, i \in U \quad z_{it} \leq z_{i,t+1}. \quad (2.10)$$

- Kill budget (cost of killing IT modules; this is part of the objective of the kill managers):

$$\forall t \in T \setminus \{t_{\max}\} \quad \sum_{i \in U} b_i (u_{it} - u_{i,t+1}) \leq K_t, \quad (2.11)$$

where the term  $u_{it} - u_{i,t+1}$  is only ever 1 when an IT module is killed — we remark that the next constraints prevent the term from ever taking value  $-1$ .

- Once an IT module is killed, cannot activate it again.

$$\forall t \in T \setminus \{t_{\max}\}, i \in U \quad u_{it} \geq u_{i,t+1}. \quad (2.12)$$

- If an existing service is active, the necessary IT modules must also be active:

$$\forall t \in T, (i, j) \in A \quad u_{it} \geq v_{jt}. \quad (2.13)$$

- If a new service is active, the necessary IT modules must also be active:

$$\forall t \in T, (i, k) \in B : i \in N \quad z_{it} \geq w_{kt}. \quad (2.14)$$

- An existing service can be deactivated once all departments relying on it have already switched to new services; for this purpose, we define sets  $\mathcal{W}_j = \{k \in W \mid \exists \ell \in D ((\ell, j) \in E \wedge (\ell, k) \in F)\}$  for all  $j \in V$ :

$$\forall t \in T, j \in V \quad \sum_{k \in \mathcal{W}_j} (1 - w_{kt}) \leq |\mathcal{W}_j| v_{jt}. \quad (2.15)$$

- New modules must be deployed according to precedences: for a precedence  $(i, h) \in S$ ,  $i$  must be deployed at least one timestep before  $h$  is; therefore, if  $z_{it} = 0$  then  $z_{hs} = 0$  for all  $s \leq t$  (2.16), and if  $t$  is the first timestep where  $z_{it} = 1$  then  $z_{ht} = 0$  (2.17):

$$\forall s \leq t \in T, (i, h) \in S \quad z_{hs} - z_{it} \leq 0 \quad (2.16)$$

$$\forall s \leq t \in T \setminus \{0\}, (i, h) \in S \quad z_{hs} + z_{it} - z_{i,t-1} \leq 1. \quad (2.17)$$

- Boundary conditions. To be consistent with the objectives of the module and kill managers, we postulate that:

- at  $t = 0$  all IT modules needed by existing services are active, all IT modules needed by new services are inactive:

$$\forall i \in U \setminus N \quad u_{i0} = 1; \quad (2.18)$$

$$\forall i \in N \quad u_{i0} = 0; \quad (2.19)$$

$$\forall i \in U \quad z_{i0} = 0; \quad (2.20)$$

$$\forall j \in V \quad v_{j0} = 1 \quad \wedge \quad \forall k \in W \quad w_{k0} = 0. \quad (2.21)$$

- at  $t = t_{\max}$  all IT modules needed by the existing services have been killed:

$$\forall i \in U \quad u_{it_{\max}} = 0. \quad (2.22)$$

These boundary conditions are a simple implementation of the objectives of module and kill managers. Similar objectives can also be pursued by adjoining further constraints to the MP, such as for example that the number of IT modules serving ES must not exceed a given amount.

The formulation above is a Binary Quadratic Program (BQP) with two objective functions. Single-objective BQPs exhibit products of binary decision variables [133]; they can either be solved directly using standard BB-based solvers [13, 85, 153] or reformulated exactly (see [107] for a formal definition of *reformulation*) to a MILP, by means of the PRODBIN reformulation [61, 108] prior to solving is with standard

MILP solvers.

**Remark 2.3.1 (Single-objective vs bi-objective formulations)** Consider the single objective formulation with objective function (2.7) only. This pursues the maximization of the business value by activating those modules that are necessary to the implementation of new services as soon as possible. The deactivation of old modules is considered as a cost, hence this formulation forces the solutions to respect a kill budget by means of constraint (2.11). The time of deactivation might influence feasibility (through constraint (2.15)) but not the solution cost. In the biobjective formulation, on the other hand, objective function (2.8) induces an anticipation of the deactivation of the useless modules. Consider the two partial solutions showed

time	module: $u_{it}$	$u_{it} - u_{i,t+1}$	$1 - u_{it}$
1	1	0	0
2	1	0	0
3	1	0	0
4	1	1	0
5	0	0	1
...	...	...	...
time	module: $u_{it}$	$u_{it} - u_{i,t+1}$	$1 - u_{it}$
1	1	1	0
2	0	0	1
3	0	0	1
4	0	0	1
5	0	0	1
...	...	...	...

Table 2.1: Left: a partial solution of the single-objective formulation. Right: effect of second objective

in Tables 2.1. We observe that  $u_{it} - u_{i,t+1}$  is positive only at step 3, while  $1 - u_{it}$  is true from the deactivation till to the end. Thus the earlier the deactivation of a module occurs, the larger the value  $\sum_{i,t} \phi_i(1 - u_{it})$  of objective (2.8) will be.

### 2.3.3 Valid cuts from implied properties

The BB method for MPs with binary variables performs a binary tree-like recursive search. At every node, a lower bound to the optimal objective function value is computed by solving a continuous relaxation of the problem. If all integral variables happen to take integer values at the optimum of the relaxation, the node is *fathomed* with a feasible optimum. If this optimum has better objective function value than the feasible optima found previously, it replaces the *incumbent*, i.e. the best current optimum. Otherwise, a variable  $x_j$  taking fractional value  $\bar{x}_j$  is selected for branching. Two subnodes of the current node are created by imposing constraints  $x_j \leq \lfloor \bar{x}_j \rfloor$  (left node) and  $x_j \geq \lceil \bar{x}_j \rceil$  (right node) to the problem. If the relaxed

objective function value at a node is worse than the current incumbent, the node is also fathomed. The step of BB which most deeply impacts its performance is the computation of the lower bound. To improve the relaxation quality, one often adjoins “redundant constraints” to the problem whenever their redundancy follows from the integrality constraints. Thus, such constraints will not be redundant with respect to the relaxation. An inequality is *valid* for a MP if it is satisfied by all its feasible points. If an inequality is valid for an MP but not for its relaxation, it is called a *valid cut*.

We shall now discuss two valid inequalities for the evolution problem. The first one stems from the following statement: *If a new service  $k \in W$  is inactive, then all existing services linked to all departments relying on  $k$  must be active*. We formalize this statement by defining the sets:

$$\forall k \in W \quad \mathcal{V}_k = \{j \in V \mid \exists \ell \in D ((\ell, j) \in E \wedge (\ell, k) \in F)\}.$$

The statement corresponds to the inequality:

$$\forall t \in T, k \in W \quad \sum_{j \in \mathcal{V}_k} (1 - v_{jt}) \leq |\mathcal{V}_k| w_{kt}. \quad (2.23)$$

**Lemma 2.3.2** *Whenever  $(v, w)$  are part of a feasible solution of the evolution problem, (2.15)  $\Leftrightarrow$  (2.23).*

Firstly, we start proving that (2.15)  $\Rightarrow$  (2.23). We proceed by contradiction: suppose (2.15) holds and (2.23) does not. Then there must be  $t \in T, k \in W, j \in \mathcal{V}_k$  such that  $w_{kt} = 0$  and  $v_{jt} = 0$ . By (2.15),  $v_{jt} = 0$  implies  $\forall h \in \mathcal{W}_j (w_{ht} = 1)$ . By definition of  $\mathcal{V}_k$  and  $\mathcal{W}_j$ , we have that  $k \in \mathcal{W}_j$ , and hence  $w_{kt} = 1$  against the assumption. Secondly, we observe that the converse, (2.15)  $\Leftarrow$  (2.23), also holds. The proof is symmetric: it suffices to swap  $j$  with  $k$ ,  $\mathcal{W}_j$  with  $\mathcal{V}_k$ ,  $v$  with  $w$ , (2.15) with (2.23).

We remark that (2.15)  $\Rightarrow$  (2.23) let us assert that (2.23) is a valid inequality for the AEP. The second inequality is a simple relation between  $v$  and  $w$ .

**Proposition 2.3.3** *The inequalities*

$$\forall t \in T, j \in V, k \in W \exists \ell \in D ((\ell, j) \in E \wedge (\ell, k) \in F) \quad v_{jt} + w_{kt} \geq 1 \quad (2.24)$$

*are valid for the AEP.*

Suppose (2.24) does not hold: hence there are  $t \in T, j \in V, k \in W, \ell \in D$  with  $(\ell, j) \in E$  and  $(\ell, k) \in F$  such that  $v_{jt} + w_{kt} = 0$ . Since  $v_{jt}, w_{kt} \geq 0$ , this implies  $v_{jt} = w_{kt} = 0$ . It is easy to verify that if this is the case, (2.15) and (2.23) cannot both hold, contradicting (2.15)  $\Leftrightarrow$  (2.23) (cf. Lemma (2.3.2)).



Eq. (2.24) states that at any given time period no pair (ES, NS) related to a given department must be inactive (otherwise the department cannot be functional). We can add (2.23) and (2.24) to the MP formulation of the AEP, and hope they will improve the quality of the lower bound obtained via the LP relaxation. This is verified empirically in Sect. 2.5.3.

## 2.4 Formulation properties and trade-off

As mentioned previously, the formulation (2.3)-(2.22) models a biobjective problem. Notice, however, that the two objective functions (2.7), (2.8) involve different sets of variables. Thus, it is not immediately evident that this formulation might not be decomposed in two separate problems, one maximizing (2.7) and the other (2.8). Nor it is evident that the Pareto region might consist of more than one single optimum. In other words, establishing whether this bi-objective formulation really corresponds to a trade-off type of problem is a relevant question.

In this section we present a mathematical analysis which shows that this is indeed the case. Specifically, we show that without (2.15), the problem can be decomposed in such a way that the only efficient solution is the utopia point. In this sense, (2.15) are the true source of the trade-off nature of (2.3)-(2.22). We first analyse a Lagrangian relaxation of (2.15), then give an example of a class of problem instances whose corresponding Pareto region fails to be a singleton set. For simplicity, we consider the original formulation only, without the valid cuts of Sect. 2.3.3.

### 2.4.1 Decomposability

Without (2.15), the formulation can be decomposed into two bi-objective subproblems involving only the  $u, v$  and respectively  $w, z$  variables. This suggests a Lagrangian relaxation [175] of (2.15) using Lagrangian coefficients  $\lambda, \mu \geq 0$ , which yields the two maximization objectives:

$$\begin{aligned} \zeta(\lambda, u, v, w, z) &= \sum_{\substack{(i,k) \in \mathcal{B} \\ t \in T}} \beta_{ik} z_{it} w_{kt} + \sum_{\substack{t \in T \\ j \in V}} \lambda_{jt} \left( \sum_{k \in \mathcal{W}_j} (w_{kt} - 1) + |\mathcal{W}_j| v_{jt} \right) = \\ &= \sum_{\substack{t \in T \\ (i,k) \in \mathcal{B}}} \beta_{ik} z_{it} w_{kt} + \sum_{\substack{t \in T, j \in V \\ k \in \mathcal{W}_j}} \lambda_{jt} w_{kt} + \sum_{\substack{t \in T \\ j \in V}} |\mathcal{W}_j| \lambda_{jt} (v_{jt} - 1), \end{aligned}$$

and

$$\begin{aligned} \eta(\mu, u, v, w, z) &= \sum_{\substack{t \in T \\ i \in U}} \phi_i u_{it} + \sum_{\substack{t \in T \\ j \in V}} \mu_{jt} \left( \sum_{k \in \mathcal{W}_j} (w_{kt} - 1) + |\mathcal{W}_j| v_{jt} \right) = \\ &= \sum_{\substack{t \in T \\ i \in U}} \phi_i u_{it} + \sum_{\substack{t \in T, j \in V \\ k \in \mathcal{W}_j}} \mu_{jt} w_{kt} + \sum_{\substack{t \in T \\ j \in V}} |\mathcal{W}_j| \mu_{jt} (v_{jt} - 1). \end{aligned}$$

Thus, we can decompose the problem of Sect. 2.3.1 into the two bi-objective Lagrangian subproblems  $P, Q$ :

$$\left. \begin{array}{l}
 \max_{u,v} \quad \sum_{\substack{t \in T \\ j \in V}} |\mathcal{W}_j| \lambda_{jt} (v_{jt} - 1) \\
 \max_{u,v} \quad \sum_{\substack{t \in T \\ i \in U}} \phi_i u_{it} + \sum_{\substack{t \in T \\ j \in V}} |\mathcal{W}_j| \mu_{jt} (v_{jt} - 1) \\
 \forall t \in T \setminus \{t_{\max}\} \quad \sum_{i \in U} b_i (u_{it} - u_{i,t+1}) \leq K_t \\
 \forall t \in T \setminus \{t_{\max}\}, i \in U \quad u_{it} \geq u_{i,t+1} \\
 \forall t \in T, (i, j) \in A \quad u_{it} \geq v_{jt} \\
 \forall i \in U \quad u_{i0} = 1 \\
 \forall j \in V \quad v_{j0} = 1 \\
 \forall i \in U \quad u_{it_{\max}} = 0,
 \end{array} \right\} \quad (2.25)$$

$$\left. \begin{array}{l}
 \max_{w,z} \quad \sum_{\substack{t \in T \\ (i,k) \in B}} \beta_{ik} z_{it} w_{kt} + \sum_{\substack{t \in T, j \in V \\ k \in \mathcal{W}_j}} \lambda_{jt} w_{kt} \\
 \max_{w,z} \quad \sum_{\substack{t \in T, j \in V \\ k \in \mathcal{W}_j}} \mu_{jt} w_{kt} \\
 \forall t \in T \setminus \{t_{\max}\} \quad \sum_{i \in U} a_i (z_{i,t+1} - z_{it}) \leq H_t \\
 \forall t \in T \setminus \{t_{\max}\}, i \in U \quad z_{it} \leq z_{i,t+1} \\
 \forall t \in T, (i, k) \in B : i \in N \quad z_{it} \geq w_{kt} \\
 \forall s \leq t \in T, (i, h) \in S \quad z_{hs} - z_{it} \leq 0 \\
 \forall s \leq t \in T \setminus \{0\}, (i, h) \in S \quad z_{hs} + z_{it} - z_{i,t-1} \leq 1 \\
 \forall i \in U \quad z_{i0} = 0 \\
 \forall j \in V \quad w_{j0} = 0.
 \end{array} \right\} \quad (2.26)$$

We remark that objective functions for (2.25) and (2.26) are specific instances of the following more general case:

$$\left. \begin{array}{l}
 \max_{y \in Y} \quad f(y) + g(y) \\
 \max_{y \in Y} \quad g(y)
 \end{array} \right\}. \quad (2.27)$$

for functions  $f, g$  and a set  $Y$ . We have the following results.

**Proposition 2.4.1** *The efficient set of (2.27) is contained in that of:*

$$\left. \begin{array}{l}
 \max_{y \in Y} \quad f(y) \\
 \max_{y \in Y} \quad g(y)
 \end{array} \right\}. \quad (2.28)$$

Let  $y, y'$  be in the efficient set of (2.27). Then, either  $f(y) + g(y) \leq f(y') + g(y')$  (\*) and  $g(y) \geq g(y')$  (†), or  $f(y) + g(y) \geq f(y') + g(y')$  (\*\*) and  $g(y) \leq g(y')$  (‡). By

rearrangement of (\*) we have  $f(y) - f(y') \leq g(y') - g(y)$ ; by (†),  $g(y') - g(y) \leq 0$ . Therefore,  $f(y) - f(y') \leq 0$ . By rearrangement of (\*\*) we have  $f(y) - f(y') \geq g(y') - g(y)$ , which by (‡) is  $\geq 0$ , hence  $f(y) - f(y') \geq 0$ . Thus  $y, y'$  are in the efficient set of (2.28).

By Prop. 2.4.1 one could find the efficient sets of:

$$\left. \begin{array}{l} \max_{u,v} \sum_{\substack{t \in T \\ j \in V}} |\mathcal{W}_j| \lambda_{jt} (v_{jt} - 1) \\ \max_{u,v} \sum_{\substack{t \in T \\ i \in U}} \phi_i u_{it} \\ \text{constraints of (2.25)} \end{array} \right\} \quad (2.29)$$

$$\left. \begin{array}{l} \max_{w,z} \sum_{\substack{t \in T \\ (i,k) \in B}} \beta_{ik} z_{it} w_{kt} \\ \max_{w,z} \sum_{\substack{t \in T, j \in V \\ k \in \mathcal{W}_j}} \mu_{jt} w_{kt} \\ \text{constraints of (2.26)} \end{array} \right\} \quad (2.30)$$

and then simply filter out all the dominated solutions with respect to the objectives of (2.25) and, respectively, (2.26).

Formulations (2.29)-(2.30) are difficult to solve because, in accordance with Lagrangian duality theory, one would also have to minimize with respect to  $\lambda, \mu$ . In practice, one could employ a “pure decomposition” where  $\lambda = \mu = 0$ . This reduces (2.29)-(2.30) to the two following single-objective problems:

$$\left. \begin{array}{l} \max_{u,v} \sum_{\substack{t \in T \\ i \in U}} \phi_i u_{it} \\ \text{constraints of (2.25)} \end{array} \right\} \quad (2.31)$$

$$\left. \begin{array}{l} \max_{w,z} \sum_{\substack{t \in T \\ (i,k) \in B}} \beta_{ik} z_{it} w_{kt} \\ \text{constraints of (2.26)} \end{array} \right\} \quad (2.32)$$

This proves the following result:

**Theorem 2.4.2** *Relaxing (2.15) yields a MP with the single objective function (2.7) + (2.8).*

In other words, the MP cannot be decomposed unless constraints (2.15) are relaxed.

## 2.4.2 Trade-off

Often, in complex environments, architects and systems engineers have to regard many non-functional requirements such as safety and maintainability. These requirements do not change the main mission of a system which is to fulfil the functional requirements but influence its quality. Different architectures could attain the target, uncovering different additional features. The main functional requirement that the evolution of an information system must satisfy is the complete switching to new services without service discontinuity. This task is optimized when the transition can be done quickly, in order to profit from the income provided by the new services. The first objective of the model (business gain) expresses this feature. The second objective, killing gain, incorporates the need of keeping the system “clean”, because it rewards the fact that old modules are removed. These useless modules pollute the system and may, in the worst case, introduce some elements of risk, if obsolete

functionalities are not managed properly. Thus, this second objective pushes the research of transitions which leads to a configuration of the whole system which exhibits good non-functional attributes.

Consider the following class of instances:  $U = \{1, 2\}$ ,  $N = \{1, 2\}$ ,  $V = \{1, 2\}$ ,  $D = \{1, 2\}$ ,  $W = \{3, 4\}$ ,  $A = \{(1, 1), (2, 2)\}$ ,  $B = \{(1, 3), (2, 4)\}$ ,  $E = \{(1, 1), (2, 2)\}$ ,  $F = \{(1, 3), (2, 4)\}$ , summarized graphically in Fig. 2.8. Two IT modules have to be

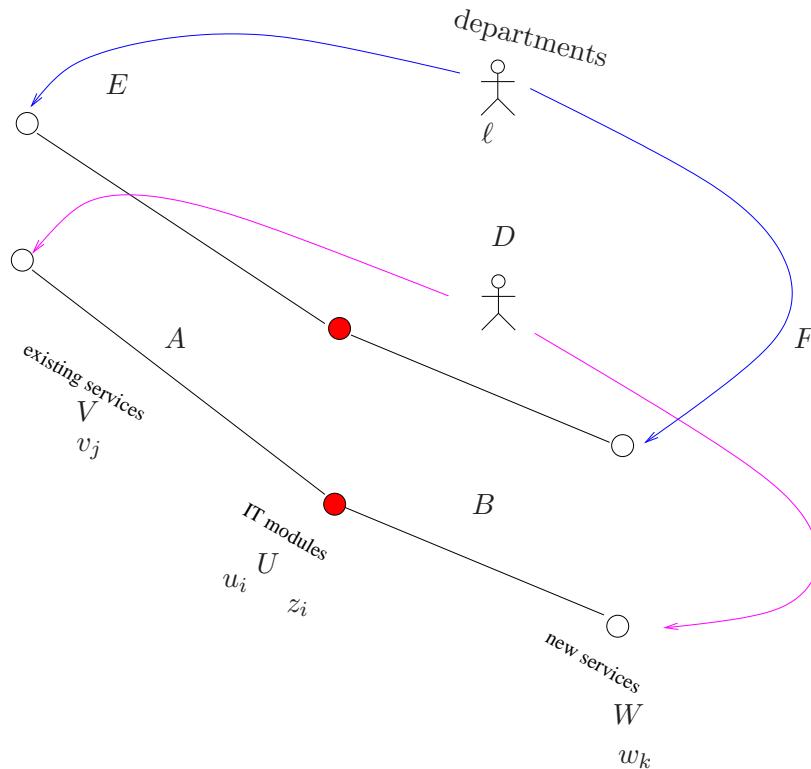


Figure 2.8: Basic instance

switched from old services to new services by activating/deactivating their interfaces to these services. The set  $T$  is limited to only two time periods. From this class, consider the specific instances given by the parameter values in Table 2.2.

$i$	$a_i$	$b_i$	$\phi_i$	$\beta_{i1}$	$\beta_{i2}$
1	1	1	2	1	0
2	1	1	1	0	2

$t$	$H_t$	$K_t$
0	0	0
1	1	1
2	1	1

Table 2.2: Time-independent (left) and time-dependent (right) parameter values

We remark on some effects produced by the constraints of the model combined with the budget thresholds in Table 2.2. The production budget constraints (2.9) prevent both new modules from being switched on at the same time period, because this would cost 2 monetary units and the allowed bound is 1. Similarly, the kill budget constraints (2.11) prevent both old modules from being switched off at the

same time period. The utopia point therefore corresponds to the solution given in Table 2.3, where the most profitable modules are switched on/off first. Because of

$t$	$u_{1t}$	$u_{2t}$	$z_{1t}$	$z_{2t}$	$v_{1t}$	$v_{2t}$	$w_{1t}$	$w_{2t}$
0	1	1	0	0	1	1	0	0
1	0	1	0	1	0	1	0	1
2	0	0	1	1	0	0	1	1

Table 2.3: Utopia point

the values of  $\phi$  and  $\beta$ , however, this solution is infeasible with respect to constraints 2.15.

These constraints require that when old modules are deactivated, and hence the corresponding services are stopped, the replacing (new) services, on which the departments rely, must already be in place. For example, if we switch the old module 1 off, the old service 1 based on it has to be halted. Thus, department 1, which loses its existing service 1, needs the new service 3, and consequently the module 1 has to be switched (and similarly for department 2 with old service 2 and new service 4 and the corresponding module 2). The partial solution  $u_1 = 0 \wedge z_2 = 1$  is not possible because of constraints (2.15), and this makes the utopical solution in Table 2.3 infeasible. The allowed partial solutions are  $u_1 = 0 \wedge z_1 = 1$  and  $u_2 = 0 \wedge z_2 = 1$ . Thus, a feasible efficient transition is represented by table 2.4: at first the most profitable deactivation is realised and then the less profitable activation must be carried out to avoid service discontinuity for the first business department. A second feasible and efficient transition is represented by table 2.5: at first the most

$t$	$u_{1t}$	$u_{2t}$	$z_{1t}$	$z_{2t}$	$v_{1t}$	$v_{2t}$	$w_{1t}$	$w_{2t}$
0	1	1	0	0	1	1	0	0
1	0	1	1	0	0	1	1	0
2	0	0	1	1	0	0	1	1

Table 2.4: Feasible Solution 1

profitable activation is realised, which means that the most profitable deactivation must be delayed. These two solutions are feasible. Nevertheless no option is clearly

$t$	$u_{1t}$	$u_{2t}$	$z_{1t}$	$z_{2t}$	$v_{1t}$	$v_{2t}$	$w_{1t}$	$w_{2t}$
0	1	1	0	0	1	1	0	0
1	1	0	0	1	1	0	0	1
2	0	0	1	1	0	0	1	1

Table 2.5: Feasible Solution 2

better than the other one. Business gain and killing gain are different if you choose

the first solution or the second one. If we simply sum them without preference, namely if we weight equally the objectives, we get the same gain with both solutions (9 monetary units), but the “composition” of the revenue varies. Figure 2.9 shows the Pareto region. We can observe that no solution dominates the other. If the

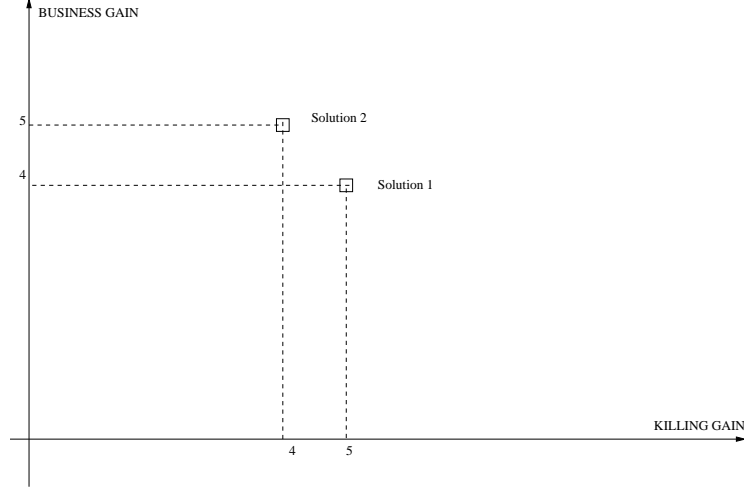


Figure 2.9: Basic trade-off

decision maker favours the deactivation of the old modules, then the first solution is preferable. If the decision maker likes the activation of the new services better, the second solution becomes more desirable. This simple example dispels the doubt that this bi-objective problem might simply be a single objective problem in disguise, and highlights constraints (2.15) as the main source of the trade-off.

## 2.5 Computational results

In [70] we proposed a single objective model of the architecture evolutions problem and showed that it can be solved in a reasonable amount of time with regard to realistically sized instances. Here, we aim to establish if we can solve the MOP involving the two objectives (2.7) and (2.8) as well. We are more interested in evaluating the computational effort required rather than in exactly modelling the preferences of the decision makers. Hence, we adopt a no-preference approach, the  $L_p$ -metric method, with  $p = 1$  and solve:

$$\min_{u,v,w,z} \left| \sum_{\substack{t \in T \\ (i,k) \in B}} \beta_{ik} z_{it} w_{kt} - h_1^* \right| + \left| \sum_{\substack{t \in T \\ i \in U}} \phi_i (1 - u_{it}) - h_2^* \right|, \quad (2.33)$$

where  $h_1^*, h_2^*$  are the optimum values of the single-objective maximizations of (2.7) and, respectively, (2.8), subject to all problem constraints.

We consider a set of small instances, to be solved to guaranteed optimality, and another set of larger instances where the BB algorithm is stopped either at BB

termination or after 30 minutes of CPU time (whichever comes first). We use the AMPL modelling environment [63] and the CPLEX 12.2 solver [85] running with its default configuration on a single 2.4 GHz Intel Xeon CPU with 8GB RAM. Ordinarily CPLEX's Quadratic Programming (QP) solver requires QPs with Positive Semi-Definite (PSD) quadratic forms only. Although in our case this may fail to hold, CPLEX can reformulate the problem exactly to the required form because all variables involved in the quadratic products are binary.

We consider the same set of instances both for the single objective form of the problem and the bi-objective form. All instances have been randomly generated from a model that bears some similarity to data coming from an actual service industry. We consider three parameter categories: cardinalities (vertex set), graph density (edge creation probability) and monetary values. Each of the 64 instances in each set corresponds to a triplet (cardinality, edge creation probability, monetary value), each component of which ranges over a set of four elements.

Since our solution method of choice, which consists in solving (2.33), makes use of the single-objective optimum values  $h_1^*$  and  $h_2^*$ , we have to compute them. We remind that our previous paper [70] deals with the CPU time necessary for solving the mono-objective model which considers only the Business Gain and which provides the values of  $h_1^*$ . Thus, we focus now on CPU times necessary for solving the mono-objective model which considers only the Killing Gain and which gives the values of  $h_2^*$ . The results, for medium and big instances, are reported in Table 2.6. We remark that it can be achieved with small computational effort.

We suppose  $h_1^*, h_2^*$  as pre-calculated in the tests we present in the next section. Hence, our subsequent results do not consider the time needed to calculate the utopia point.

### 2.5.1 CPU time

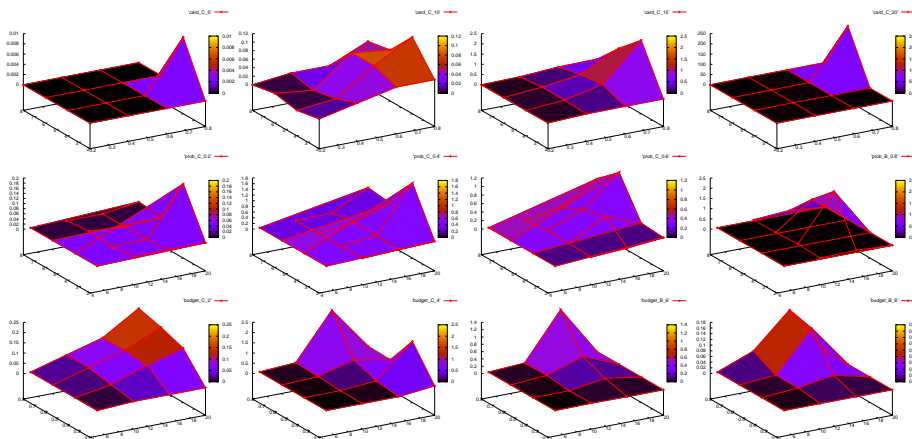


Figure 2.10: Bi-objective Model: CPU time when solving small instances.

Card.	Prob.	Bud.	feas.	cpu	Killing Gain
25	0.2	10	feasible	0.01	300
25	0.2	12	feasible	0.01	300
25	0.2	14	feasible	0.01	300
25	0.2	16	feasible	0.01	300
25	0.4	10	feasible	0.74	159
25	0.4	12	feasible	0.06	225
25	0.4	14	feasible	0.07	225
25	0.4	16	feasible	0.07	228
25	0.6	10	feasible	0.43	150
25	0.6	12	feasible	0.43	150
25	0.6	14	feasible	0.10	225
25	0.6	16	feasible	0.12	225
25	0.8	10	feasible	0.24	150
25	0.8	12	feasible	0.21	150
25	0.8	14	feasible	0.12	225
25	0.8	16	feasible	0.14	225
30	0.2	10	feasible	0.06	315
30	0.2	12	feasible	0.01	360
30	0.2	14	feasible	0.13	327
30	0.2	16	feasible	0.01	360
30	0.4	10	feasible	0.53	183
30	0.4	12	feasible	0.53	180
30	0.4	14	feasible	0.13	270
30	0.4	16	feasible	0.11	270
30	0.6	10	feasible	0.45	180
30	0.6	12	feasible	0.43	180
30	0.6	14	feasible	0.49	180
30	0.6	16	feasible	0.16	270
30	0.8	10	feasible	0.41	180
30	0.8	12	feasible	0.46	180
30	0.8	14	feasible	0.43	180
30	0.8	16	feasible	0.18	270
35	0.2	10	feasible	0.02	420
35	0.2	12	feasible	0.07	354
35	0.2	14	feasible	0.18	354
35	0.2	16	feasible	0.02	420
35	0.4	10	feasible	3.83	111
35	0.4	12	feasible	0.58	210
35	0.4	14	feasible	0.51	210
35	0.4	16	feasible	2.03	213
35	0.6	10	feasible	0.98	105
35	0.6	12	feasible	0.47	210
35	0.6	14	feasible	0.52	210
35	0.6	16	feasible	0.46	210
35	0.8	10	feasible	0.73	105
35	0.8	12	feasible	0.52	210
35	0.8	14	feasible	0.51	210
35	0.8	16	feasible	0.45	210
40	0.2	10	feasible	0.10	369
40	0.2	12	feasible	0.07	396
40	0.2	14	feasible	0.12	381
40	0.2	16	feasible	0.02	480
40	0.4	10	feasible	0.98	120
40	0.4	12	feasible	3.76	120
40	0.4	14	feasible	3.66	240
40	0.4	16	feasible	1.19	240
40	0.6	10	feasible	1.31	120
40	0.6	12	feasible	1.64	120
40	0.6	14	feasible	0.58	240
40	0.6	16	feasible	0.80	240
40	0.8	10	feasible	0.88	120
40	0.8	12	feasible	1.71	120
40	0.8	14	feasible	0.82	240
40	0.8	16	feasible	0.89	240

Table 2.6: Killing Gain - Single Objective

In order to observe how CPU time scales when solving to guaranteed optimality, we present 12 plots referring to the small set, grouped by row. We plot seconds of user CPU time: for each fixed cardinality, in function of edge creation probability and monetary value (Fig. 2.10, first row); for each fixed edge creation probability, in function of cardinality and monetary value (Fig. 2.10, second row); for each fixed monetary value, in function of cardinality and edge creation probability (Fig. 2.10, third row). The largest “small instance” corresponds to the triplet (20, 0.8, 8). The plots show that the proposed methodology can solve a small instance to guaranteed optimality within half an hour on standard computational equipment; it is also possible to notice that denser graphs and smaller budgets yield more difficult in-



stances. Sudden drops in CPU time might correspond to infeasible instances, which, interestingly, are usually detected quite fast.

These results and the results reported in [70] show that we can solve both the single objective and the bi-objective formulations for realistic instances reasonably quickly. Table 2.7 shows the results of the comparison (with cardinality fixed at 20, i.e. the largest instances in the set of small instances) and the relative increase of the CPU time needed to solve the bi-objective formulation, with respect to the single-objective one (which considers only Business Gain). The effort is considerably higher but still manageable for practical purposes. Infeasibility is detected similarly in both models. Table 2.8 reports the results for medium and big instances. However in this case the timeout is set to 30 minutes and thus only executions shorter than 30 minutes are relevant to the comparison of CPU time. The other executions are relevant only for the evaluation of the optimality gap.

Edge Probability	Budget	feasible/infeasible	CPU time increment
0.4	2	infeasible	0.0
0.4	4	feasible	314.3
0.4	6	feasible	241.2
0.4	8	feasible	257.9
0.6	2	infeasible	12.5
0.6	4	infeasible	0.0
0.6	6	feasible	73.4
0.6	8	feasible	135.7
0.8	2	infeasible	-4.5
0.8	4	infeasible	4.0
0.8	6	feasible	186.7
0.8	8	feasible	137.0

Table 2.7: CPU time increment

## 2.5.2 Optimality Gap

In Fig. 2.11 and Fig. 2.12 we plot the *optimality gap* at termination (which is limited to 30 minutes). The largest “large instance” corresponds to the triplet (40, 0.8, 16). The optimality gap, expressed in percentage, is defined by CPLEX as  $\left(\frac{100|f^* - \bar{f}|}{|f^* + 10^{-10}|}\right)\%$ , where  $f^*$  is the objective function value of the best feasible solution found within the time limit, and  $\bar{f}$  is the tightest overall lower bound. A gap of 0% corresponds to the instance being solved to optimality. The plots show that the proposed methodology is able to solve large instances to a gap of 12.8% within half an hour of CPU time at worst. It can solve and to an average gap of 1.13% both the single objective (which considers only Business Gain) and the bi-objective formulation (which considers both Business Gain and Killing Gain), within an average CPU time of 459s and 538s respectively (about 8 minutes). Table 2.8 reports the details of this comparison.

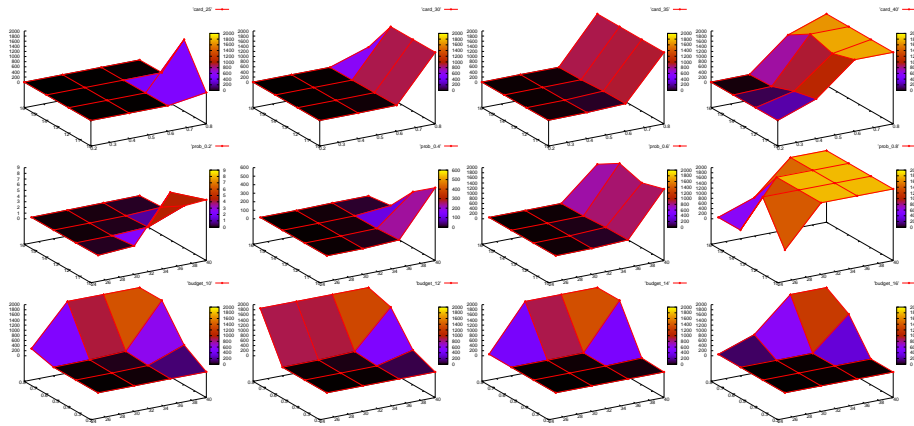


Figure 2.11: Single Objective Model: Optimality gap.

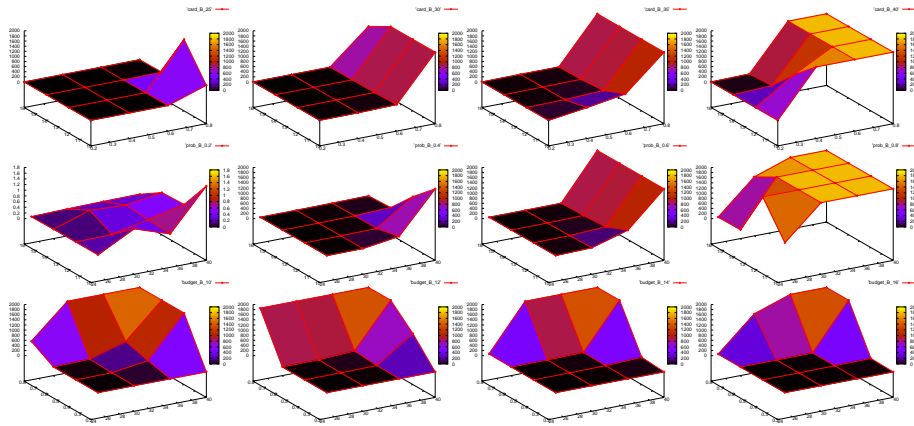


Figure 2.12: Bi-objective Model: Optimality gap.

### 2.5.3 Cuts effectiveness

As discussed in Section 2.3.3, valid cuts are redundant for the original formulation, but may improve the bound given by its continuous relaxation. In order to show that the cuts we have introduced are actually useful, we compare the optimal solution value of the continuous relaxation with and without the cuts.

Table 2.9 reports the most interesting variations of the objective function we recorded during the tests. The value of the average variation for all the instances is 0.0106. Although this may not sound so impressive, one must bear in mind that these values refer to the root node relaxation only: improvements in deeper BB nodes might improve the bound considerably. Table 2.9 should only be taken to be a counterexample dispelling the doubt that our cuts might be supposed useless.

### 2.5.4 Trade-off in realistic instances

We empirically observed a trend involving the edge density of the tripartite graph  $(D \cup V \cup W, E \cup F)$  and the shape of the Pareto region. It is easy to see that, if

Card.	Prob.	Bud.	feas. (1)	CPU (1)	obj. (1)	gap (1)	feas. (2)	CPU (2)	obj. (2)	gap (2)	CPU time in- crement
Single Objective Model						Bi-objective Model					
25	0.20	10	yes	0.09	230	0.13	yes	0.03	529	0.37	-66.6
25	0.20	12	yes	0.03	261	0.58	yes	0.04	561	0.23	33.3
25	0.20	14	yes	0.02	190	0.65	yes	0.03	490	0.25	50.0
25	0.20	16	yes	0.02	205	0.06	yes	0.02	505	0.02	0.0
25	0.40	10	yes	0.12	344	1.45	yes	0.69	497	0.63	475.0
25	0.40	12	yes	0.17	404	0.21	yes	0.35	629	0.84	105.8
25	0.40	14	yes	0.04	411	1.62	yes	0.26	636	1.25	550.0
25	0.40	16	yes	0.1	424	3.41	yes	0.44	652	0.3	340.0
25	0.60	10	yes	1.33	424	0.42	yes	3.14	574	0.03	136.0
25	0.60	12	yes	0.46	499	0.37	yes	2.48	649	0.76	439.1
25	0.60	14	yes	0.22	578	1.13	yes	0.64	803	0.03	190.9
25	0.60	16	yes	0.32	562	0.79	yes	0.72	787	0.44	125.0
25	0.80	10	yes	196.76	540	0.01	yes	504.4	690	0.01	156.3
25	0.80	12	yes	1801.15	631	0.55	yes	1800.55	781	0.83	0.0
25	0.80	14	yes	5.85	734	0.02	yes	13.4	959	0.01	129.0
25	0.80	16	yes	2.15	745	0.08	yes	6.33	970	0.04	194.4
30	0.20	10	yes	0.51	304	0.05	yes	0.69	612	1.04	35.2
30	0.20	12	yes	0.05	344	0.59	yes	0.05	704	0.24	0.0
30	0.20	14	yes	0.04	303	1.05	yes	0.44	627	0.21	1000.0
30	0.20	16	yes	0.06	315	0.53	yes	0.05	674	0.47	-16.6
30	0.40	10	yes	2.54	477	0.09	yes	7.34	657	0.06	188.9
30	0.40	12	yes	0.95	532	0.14	yes	4.36	712	0.11	358.9
30	0.40	14	yes	0.27	609	0.49	yes	0.76	879	0.89	181.4
30	0.40	16	yes	0.26	560	0.16	yes	1.94	830	0.06	646.1
30	0.60	10	yes	13.27	635	0.01	yes	37.76	815	0.01	184.5
30	0.60	12	yes	10.06	634	0.02	yes	11.43	814	0.07	13.6
30	0.60	14	yes	3.11	734	0.06	yes	5.81	914	0.11	86.8
30	0.60	16	yes	0.31	824	0.19	yes	1.2	1094	0.34	287.1
30	0.80	10	yes	1800.96	772	3.84	yes	1800.46	952	3.55	0.0
30	0.80	12	yes	1800.96	789	1.31	yes	1800.45	969	1.44	0.0
30	0.80	14	yes	1800.99	837	1.73	yes	1800.46	1017	1.78	0.0
30	0.80	16	yes	359.42	1099	0.01	yes	1296.32	1369	0.01	260.6
35	0.20	10	yes	7.51	352	0.02	yes	0.28	764	0.02	-96.27
35	0.20	12	yes	0.31	377	0.04	yes	0.33	716	0.07	6.45
35	0.20	14	yes	0.09	425	0.11	yes	0.39	769	0.11	333.3
35	0.20	16	yes	0.09	437	0.03	yes	0.07	857	0.32	-22.2
35	0.40	10	yes	20.45	534	0.03	yes	212.1	641	0.02	937.1
35	0.40	12	yes	6.86	665	0.01	yes	20.34	875	0.01	196.5
35	0.40	14	yes	5.46	726	0.05	yes	15.19	936	0.04	178.2
35	0.40	16	yes	5.5	701	0.02	yes	10.72	914	0.07	94.9
35	0.60	10	yes	61.68	613	0.02	yes	459.1	718	0.01	644.3
35	0.60	12	yes	55.69	824	0.01	yes	87.78	1034	0.01	57.6
35	0.60	14	yes	12.79	816	0.02	yes	28.15	1026	0.01	120.0
35	0.60	16	yes	2.53	1012	0.13	yes	8.29	1222	0.03	227.6
35	0.80	10	yes	1800.83	579	7.74	yes	1800.45	684	7.32	0.0
35	0.80	12	yes	1800.85	978	4.97	yes	1800.4	1188	4.5	0.0
35	0.80	14	yes	1800.87	969	2.45	yes	1800.45	1179	2.33	0.0
35	0.80	16	yes	1800.72	1121	0.31	yes	1800.37	1331	0.56	0.0
40	0.20	10	yes	6.13	463	0.01	yes	1.72	812	0.04	-71.9
40	0.20	12	yes	2.75	453	0.01	yes	0.62	837	0.04	-77.4
40	0.20	14	yes	0.39	449	0.03	yes	0.59	822	0.01	51.2
40	0.20	16	yes	0.19	479	0.16	yes	0.08	948	0.1	-57.8
40	0.40	10	yes	511.06	588	0.01	yes	1800.31	708	1.38	252.2
40	0.40	12	yes	324.62	686	0.01	yes	994.54	806	0.01	206.3
40	0.40	14	yes	41.64	818	0.01	yes	69.13	1058	0.01	66.0
40	0.40	16	yes	7.48	906	0.03	yes	25.12	1146	0.02	235.8
40	0.60	10	yes	1800.88	638	2.36	yes	1800.33	758	6.72	0.0
40	0.60	12	yes	1273.08	753	0.01	yes	1800.27	873	2.9	41.4
40	0.60	14	yes	1800.95	1061	0.73	yes	1800.37	1301	0.66	0.0
40	0.60	16	yes	1222.44	1105	0.01	yes	1800.36	1345	0.37	47.2
40	0.80	10	yes	1800.72	720	12.84	yes	1800.33	840	11.49	0.0
40	0.80	12	yes	1800.75	807	9.18	yes	1800.46	927	8.3	0.0
40	0.80	14	yes	1800.72	1340	6.08	yes	1800.39	1580	5.48	0.0
40	0.80	16	yes	1800.74	1315	3.59	yes	1800.53	1555	3.41	0.0

Table 2.8: Comparison

Card.	Prob.	Bud.	Objective function variation
25	0.4	10	0.012
30	0.2	10	0.016
35	0.2	12	0.011
40	0.2	10	0.018
40	0.2	14	0.019

Table 2.9: Cuts effectiveness

$E \cup F = \emptyset$ , then constraints (2.15) disappear, and hence the efficient set only consists of the utopia point. Pareto regions of different shapes and sizes can be obtained by employing instances with different edge sets  $E \cup F$ .

We consider medium-sized realistic instances (which correspond to triplets  $(30, p, b(p))$ , where vertex cardinality is fixed, the edge probability changes, and the budget is augmented with respect to the edge probability) and perform computational tests using the weighted-sum method. Varying the  $\alpha$  coefficient vector, we obtain different points in the Pareto region. Fig. 2.13 shows three different Pareto regions, corre-

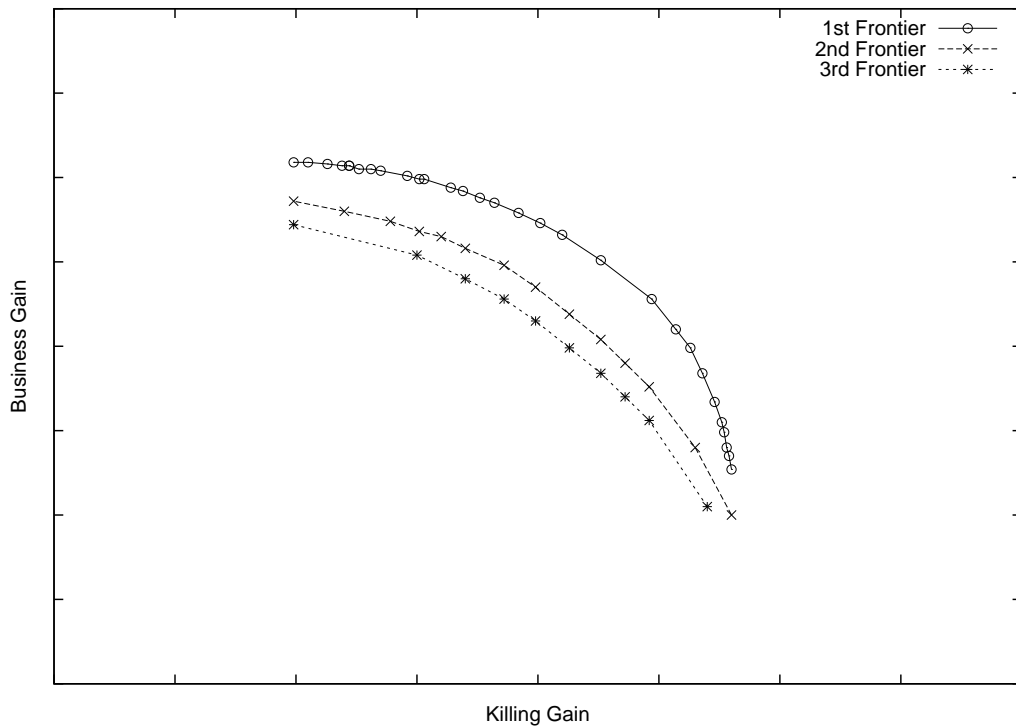


Figure 2.13: Pareto frontiers for realistic instances

sponding to three different densities of the bipartite graphs linking departments with old and new services: densest graphs yield flatter Pareto regions, and vice versa.

The architecture of an information system is done to fulfil a set of requirements expressed by several stakeholders. When new stakeholders intervene or when new needs arise, as a consequence, this set of requirements change. The architecture of the information system must evolve consistently. We have showed as the management of the transition from one architecture to another can be formulated and solved as an optimization problem that integrates high-level goals. This application shows the viability of this approach in a realistic scenario. In Section 6 we present some general conclusions.



# The recommendation problem

## 3.1 Introduction

The amount of available sources of information has strongly increased during the last decades. The overwhelming growth of the Web is probably the clearer example. From one hand this is a wonderful progress but on the other hand this poses problems, because it is not easy to retrieve the relevant information when required. Too much information can puzzle users and decision makers. Thus, the need for new technologies that can help search and retrieve information is huge. Recommender systems (RS) are one of these technologies. RS exploit a set of established user preferences to predict items<sup>1</sup> (topics, movies, books, restaurants. . .) that a user might like. RS are designed to help users, who lack knowledge to evaluate the high number of alternatives provided by several sources, first of all the Web. Recommender Systems (RS) have become an important research area and emerged as an independent field in the mid-1990s [73, 27, 3]. This research field has its roots in Information Retrieval (IR). RS can be seen as a kind of search engine. Nevertheless, important differences can be noticed. The main one is that recommender systems aim to provide personalized recommendations. IR develops global retrieval methods, but does not focus on individual needs and preferences of users. Basically, this is true also for “traditional” search engines. Moreover, search engines answer to questions. Users have to provide a query or a keyword. RS do not require from users an explicit search activity, since suggestions are provided by RS in background. Thus, RS are able to suggest items before a user asks for them.

*“The Web [. . .] is leaving the era of search and entering one of discovery. What’s the difference? Search is what you do when you’re looking for something. Discovery is when something wonderful that you didn’t know existed, or didn’t know how to ask for, finds you.”<sup>2</sup>*

---

<sup>1</sup>A recommender system often is specialized in recommending a specific type of item, for example movies, books, music . . . . However, item is used as general term in the related literature.

<sup>2</sup>CNN Money, The race to create a ‘smart’ Google By Jeffrey M. O’Brien, Fortune writer.

Probably, the most famous example of recommender systems is represented by Amazon.com [109]. Users who purchase a product, for example a book, on their website, receive suggestions about similar books they might like. Similar services are provided by YouTube, Netflix, Yahoo, Tripadvisor among others.

The information that RS exploit can be of several different types (textual, users' behaviour, ratings ...). Preferences that can be explicitly expressed (typically ratings for products) or implicitly expressed (for example a recommender system can consider the visit of a page concerning a specific item in a website, as a sign of interest or positive appreciation for that item). Moreover users usually react to the first recommendations that the system provides, accepting, refusing and rating them, and these feedbacks can be used to improve the following suggestions.

Herlocker et al. provide an exhaustive list of the aims of existing RS, in their work [82].

- Find some good items: this is the most common task
- Find all good items: this task is common when the number of items is small
- Annotation in context: in an existing context, for example a list of TV shows, underline the ones that might be interesting
- Recommend a sequence: recommend a sequence of items instead of a single item (for example a play list of musical tracks)
- Recommend a bundle: propose a set of items that are complementary (for example a set of places in a travel plan)
- Just browsing: this task consists in making the browsing of a set of items (for example a catalog) as comfortable as possible
- Find a credible recommender: allow users to “play” with the recommender system to test it
- Express self: some users desire a system that helps them to express their opinion about items. A recommender system shall ease this desire
- Help others: some users want to provide information about items they tested in order to help other people.
- Influence others: some users want to persuade other people about their opinion on an item. A recommender system should prevent this purpose (but it depends on the point of view of the stakeholders of the system)

From our point of view, the first two of these possible tasks are the most common. RS are usually required to perform two jobs. On one hand, they provide the list of the best  $n$  items with reference to a user. RS community refers to this kind of suggestion

with the term *top-n recommendation*. On the other hand, they forecast the rating of given user about a given item. This kind of suggestion is called *prediction*. These two tasks are related. If you can provide predictions for each item and each user, you can also provide a top- $n$  recommendation. Nevertheless, in some situations, it is not suitable to calculate all the predictions and RS provide simply a list without a prediction about ratings. For example, the list of the most downloaded songs of a genre of music that a user likes.

### 3.1.1 Recommendation problem

In this work, we focus on the capability of RS in providing predictions. The task we deal with is the prediction of the rating of a selected user (the *active user*) for a given item (the *target item*) . We can formulate our problem as follows. We have:

- a set  $U$  of users,  $|U| = n$ ;
- a set  $I$  of items (movies, songs, restaurants...),  $|I| = m$ ;
- a gain function  $G$  which expresses the utility of an item for a user

Utility is expressed by a numeric value representing a rating (the higher the better) varying on a chosen interval  $\Psi$ , more formally the function  $G$  is defined as:

$$G: U \times I \rightarrow \Psi$$

We want to maximize the users' utility by recommending good items and advising against bad ones. The problem is that we do not know "a priori" all the values of  $G$ , hence we have to predict users' ratings.

### 3.1.2 Recommendation methods

There are several possible approaches to the design of a recommender system. The most relevant methods are the following.

- Content-based: this kind of recommender analyzes items and is able to identify its main features [9, 114, 46, 157]. Typically, items are described by documents or textual information, in which the system can browse. Consequently, items are classified by genre and typology. Users who purchase or rate positively an item that belong to a genre, receive recommendations that concern items of the same genre. For example, a content-based recommender system that suggests movies would propose science fiction ones to someone who purchased "Star Trek: The Motion Picture". This is a *white box* approach since the system builds profiles of both items and users and it knows the items it recommends. This means that, the system can tell which is the genre of an item, for example Italian, expensive, trendy if it proposes restaurants and which are the tastes of a user. Others approaches, especially collaborative filtering do not gather



all this information. Content-based recommender systems are accurate but the analysis phase, which collects information and produces items and users profile, can be long. Moreover, they fail in suggesting items that can surprise the users. They tend to recommend items that are, somehow, expectable. Again, if a user purchased “Star Trek: The Motion Picture” then “Star Trek II: The Wrath of Khan” “Star Trek III: The Search for Spock” and also “Star Wars” are probably correct recommendations but they are quite obvious (in others words, probably, the user does not need a recommender systems for them).

- Collaborative filtering: this is, probably, the most widespread approach in RS [94, 147, 158, 165, 102]. Collaborative filtering oriented recommender systems look for users who have a similar behavior, typically who rated similarly a common set of items. The underlying idea is that these common ratings correspond to an affinity of taste. Once users are clustered in homogeneous groups (called neighborhoods) the system can recommend to a first user items that a second user, who belongs to his group, likes and the first user has not already rated. If they agreed in the past for some items, probably they agree even for new items. This is a *black box* approach, since the system ignores the features of the items. Items and users are described by the ratings they receive and express, but there is not information about their nature. If a system recommends movies, for example, it only knows that “Star Trek: The Motion Picture” has been rated “good”, “very good” and “not very good” by “Fabio”, “John” and “Alice” but it can not define the genre of the movie. This is both a drawback and an asset. It is a drawback because there is a lack of information. It is an asset because this kind of system can work for all types of items, with no changes. A collaborative filtering based system that works fine with movies will work fine with restaurants and books. The system needs only ratings. Other types of RS are more dependent on the type of item they are required to suggest.
- Community-based (or Social): this type of system exploits the social network of a user [72, 34, 76, 77]. It is conceptually similar to collaborative filtering but the neighborhood is based on explicit friendships. The user chooses his friends and the system uses their ratings and opinions in order to provide suggestions. Social recommender systems are based on the assumption that friends have similar preferences and that users trust friends more than unknown users.
- Knowledge-based: this approach is based on a huge amount of information on a specific domain [32, 30]. In particular the knowledge base of the system contains information about how each item meets users needs. The system has these data from the beginning and it does not perform an analysis of the

items to gather them. It recommends items and products through an inference about a user's needs and features of the items. The drawback of this kind of recommender is that its knowledge base shall be completed by experts and, as a consequence, this process can be long and expensive.

- Hybrid recommender systems: are based on the combination of the methods we mentioned above [33].

Despite all the efforts, RS still need further development and more advanced recommendation modelling methods. Therefore, first of all we define which is our goal and in the following sections we present three different models of recommender system, a methodology to test them and the results of their comparison (even if the definition of the right way to measure the effectiveness of a recommender system is still an unsolved problem). Section 3.3 is dedicated to a combinatorial optimization based RS [149], section 3.4 to a clusters modularity based RS [148] and section 3.5 to an Information Retrieval based RS [44].

## 3.2 Operational model of a recommender system

The operational context of a recommender system includes several stakeholders. We focus on four main types.

- Basic users: they do not know the domain of application, namely the kind of items that the system can recommend and looks for suggestion that are simple to understand and appear trustworthy. They can become regular users (and, potentially, advanced users) if the service provided is good and friendly.
- Advanced users: they know very well the domain of application and looks for recommendation that are surprising, highly specialized and accurate. If the system satisfies them, they become regular customer thus they represent the main target (from a marketing point of view) of the management.
- Website administrators: they worries about the global performance of the website (most of all, about response time) and look for fast algorithms.
- Management: they need some kind of evidence that shows that the recommender system works properly in order to persuade investors and sponsors and want to attract regular customers (since these one accept more easily to create a profile, to register themselves providing some personal data and purchase "premium" services ).

Notice, that we suppose that the recommender system works as part of a website (thus we considers the website administrator. Fig. 3.1 depicts the related actor diagram that shows the main stakeholders (in *Tropos* like style).

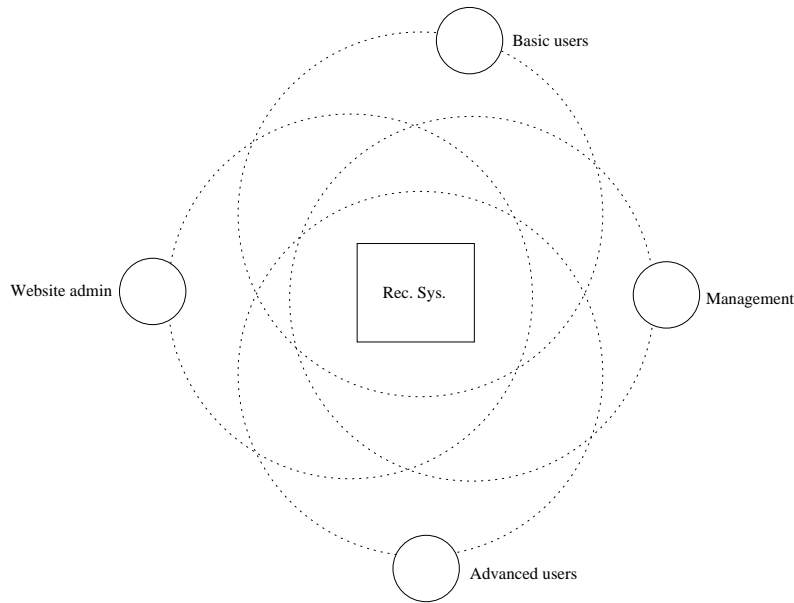


Figure 3.1: Operational context

### 3.2.1 Global Goal

Providing recommendations is part of the aims of a recommender system. The general goal we deal with is the following one:

- The recommender system shall provide recommendations to users, about items of a given domain, and it shall be good, interesting and accurate.

Notice that, the non-behavioral requirements that qualify recommender systems are numerous and some of them are non clearly defined. Among them, we consider two in particular: *be interesting* and *be accurate*. Fig. 3.2 shows the related goal diagram.

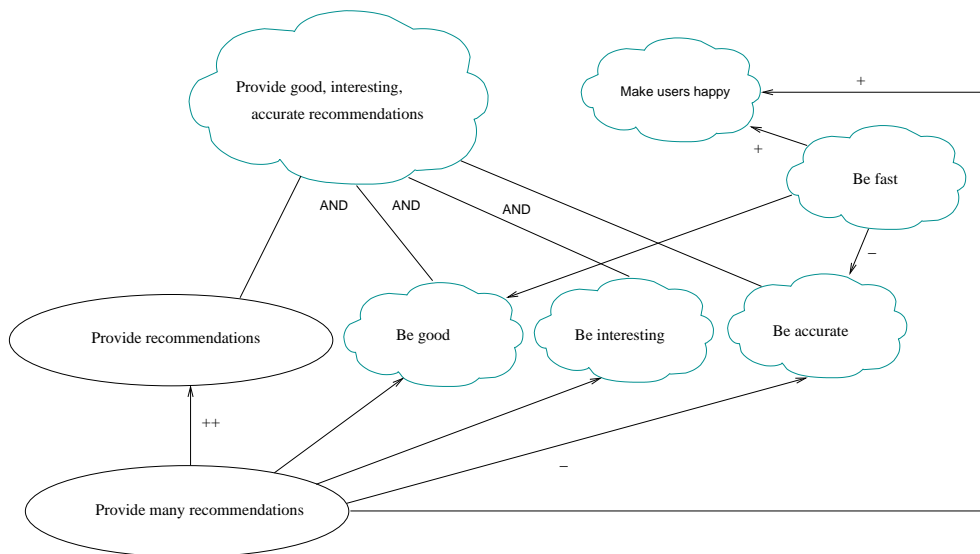


Figure 3.2: Goal Diagram

In the next sections we describe three new different recommend systems that try to get these goals.

### 3.3 Recommender system 1: TMW

The first recommender system we present, called TMW, is an hybrid one, as it is collaborative filtering oriented and community-based. It takes into account, past ratings of the active user and his known social network. If the latter is totally unknown, TMW reduces to a pure collaborative filtering system. In TMW we put together the use of a graph theoretical model and that of combinatorial optimization methods. We encode known relations between users and items and users and other users by means of weighted graphs. We then define essential components of the system by means of a combinatorial optimization problem.

#### 3.3.1 Formal model

We employ the usual graph-theoretical notation, e.g. for a vertex  $v$  of a graph  $G$ ,  $\delta_G^+(v), \delta_G^-(v)$  are the set of vertices adjacent to incoming and respectively outgoing arcs. For vertices  $u, v$  of  $G$  we also let  $\Delta_G(u, v) = \delta_G^+(u) \cap \delta_G^-(v)$ .

We are given two finite sets  $U$  (the users) and  $P$  (the items), and a vertex set  $V = U \cup P$ . We are also given two directed graphs as follows.

- A ratings bipartite digraph  $R = (V, A)$  where  $A \subseteq U \times P$  is weighted by a function  $\rho : A \rightarrow [-1, 1]$ , which expresses the ratings of users with respect to the items.
- A social network  $S = (U, B)$  weighted by a function  $\gamma : B \rightarrow [0, 1]$  which encodes a confidence coefficient between users (the *trust*).

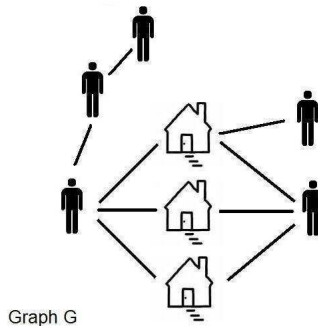


Figure 3.3: The graph  $G$ .

The union of the two graphs,  $G = R \cup S$ , is a mixed ratings/social network. Starting from  $G$  we build a new graph  $G'$  adding new arcs in  $U \times U$  or changing the values that  $\gamma$  takes on existing arcs: a missing relation of confidence between two users can

be established if both like (almost) the same items in (almost) the same way. Moreover, even when a confidence relation is already part of  $B$ , its strength can change according to similar shared preferences situations. More precisely, the reformulated graph  $G'$  is described below.

We define a graph  $G'$  with vertex set  $V' = U \cup P$  and arc set  $B'$  (weighted by a function  $\gamma' : B' \rightarrow [0, 1]$ ) defined in the following way.

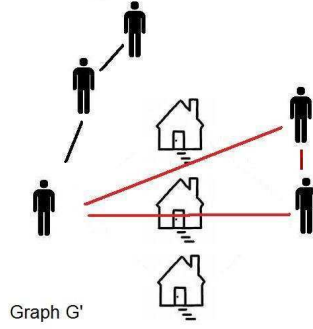
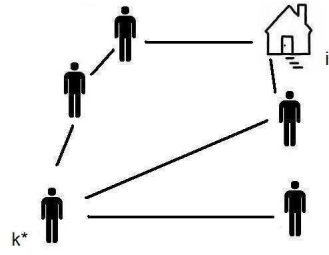


Figure 3.4: The graph  $G'$ .

1. New arcs: for every couple of users  $k, \ell \in U$  such that  $(k, \ell) \notin B$  (i.e. there is no trust arc between them) if there is a subgraph  $H = (V_H, A_H)$  of  $R$  induced by the vertex set  $V_H = \{k, \ell\} \cup \Delta_R(k, \ell)$  such that  $A_H \neq \emptyset$  (i.e. the users have common rated items), then we add to  $B'$  the arc  $(k, \ell)$  weighted by  $\gamma'_{k\ell} = f(\vartheta)$ , where  $\vartheta$  represents the difference between users computed as follow:  $\vartheta = \frac{1}{|\Delta_R(k, \ell)|} \sum_{i \in \Delta_R(k, \ell)} |\rho_{ki} - \rho_{\ell i}|$ .  $\gamma'_{k\ell}$  is obtained as a function  $f$  of  $\vartheta$ ,  $f(\vartheta) = \frac{1}{\epsilon + \vartheta}$  where  $\epsilon$  is relatively small (the exact value depends on the versions of TMW; in the version we consider in this work is 0.001). The bigger is the difference, the lower the confidence  $\gamma'_{k\ell}$ <sup>3</sup>.
2. Existing arcs: for every couple of users  $k, \ell \in U$  such that  $(k, \ell) \in B$  (i.e. there is a trust arc between them) if there is a subgraph  $H = (V_H, A_H)$  of  $R$  induced by the vertex set  $V_H = \{k, \ell\} \cup \Delta_R(k, \ell)$  such that  $A_H \neq \emptyset$  (i.e. the users have common rated items) then  $B'$  still contains the arc  $(k, \ell)$ , with a new weigh given by  $\gamma'_{k\ell} = g(\gamma_{k\ell}, f(\vartheta))$  where  $f(\vartheta)$  is defined as above and  $g$  is a weighted mean (weights depend on the version of TMW; in this version we consider here they are equal).

We let  $X = (U \times P) \setminus A$ , (i.e. not rated items) be the set of all recommendations that the system is supposed to be able to make.



Graph Z

Figure 3.5: The graph  $Z$ .

### 3.3.2 Identification of maximum confidence paths

Given a pair composed by an active user and a target item not already rated,  $(k^*, i^*) \in X$ , we consider the graph  $Z = (W, C)$  where  $W = U \cup \{i^*\}$  and  $C = B' \cup \{(k, i^*) \mid k \in \delta_R^-(i^*)\}$  (in practice, the set of neighbors of the active users and the ratings/arcs towards the target item). The ratings of the neighbors about  $i^*$  are the basic elements that we need in order to produce our prediction. The ratings of the most reliable neighbors have to be trusted more than the ones of distant neighbors. Thus, our aim is to compute a ranking for these known ratings  $\{\rho_{ki^*} \mid k \in \delta_R^-(i^*)\}$ . We exploit the confidence relations encoded in the network  $Z$ , using paths (or sets of arcs) ensuring maximum confidence. Notice that, by convention, we extend the confidence function  $\gamma$  to arcs in  $C$  adjacent to  $i^*$  as follows:  $\forall k \in \delta_R^-(i^*)$  ( $\gamma_{ki^*} = 1$ ).

Neighbors are linked to the active user by paths. In order to establish the reliability of a neighbor we have to estimate the “quality” of the path. We make the assumption that for a path  $p \subseteq C$  in  $Z$ ,  $\gamma(p) = \min_{(k,\ell) \in p} \gamma_{k\ell}$ , i.e. that the confidence on a path is defined by the lowest confidence arc in the path. This implies that finding the maximum confidence path between  $k^*$  and  $i^*$  is the same as finding a path whose arc of minimum weight  $\gamma$  is maximum (among all paths  $k^* \rightarrow i^*$ ). Considering  $Z$  as a network where  $\gamma$  are capacities on the arcs, a maximum confidence path is the same as a *maximum capacity path* between  $k^*$  and  $i^*$ , for which there exists an algorithm linear in the number of arcs [139]. The mathematical programming formulation for

<sup>3</sup>Notice that there are several techniques for calculating similarities between users (see [82] for details)

the MAXIMUM CAPACITY PATH (MCP) problem is:

$$\left. \begin{array}{ll} \max_{x,t} & t \\ \text{s.t.} & \sum_{\ell \in \delta_R^+(k^*)} x_{k^*\ell} = 1 \\ \forall \ell \in W \setminus \{k^*, i^*\} & \sum_{h \in \delta_R^-(\ell)} x_{h\ell} = \sum_{h \in \delta_R^+(\ell)} x_{\ell h} \\ \forall (k, \ell) \in C & t \leq \gamma_{k\ell} x_{k\ell} + M(1 - x_{k\ell}) \\ x \in \{0, 1\}, t & \geq 0, \end{array} \right\} \quad (3.1)$$

where  $M \geq \max_{(k,\ell) \in C} \gamma_{k\ell}$ .

Let  $\bar{p} \subseteq C$  be the maximum confidence path and  $\alpha(\bar{p}) = \operatorname{argmin}\{\gamma_{k\ell} \mid (k, \ell) \in \bar{p}\}$ . This path leads us to the most trusted neighbor. We can iterate the process, to find other ones. In fact, removing  $\alpha(\bar{p})$  from  $C^1 = C$  yields a different set of arcs  $C^2$  with associated network  $Z^2 = (W, C^2)$ , in which we can re-solve (3.1) to obtain a path  $\bar{p}^2$  as long as  $Z^2$  is connected (otherwise, define  $\bar{p}^2 = \emptyset$ ): this defines an iterative process for obtaining a sequence of triplets  $(Z^r, \bar{p}^r, r)$ . Given a confidence threshold  $\Gamma \in [0, 1]$  and an integer  $q > 0$ , we define the set  $\Omega = \{\bar{p}^r \mid \bar{p}^r \neq \emptyset \wedge r \leq q \wedge \gamma_{\alpha(\bar{p}^r)} \geq \Gamma\}$  of the  $q$  high confidence paths from  $k^*$  to  $i^*$ .

### 3.3.3 Exploiting the ratings

Recall that each  $p \in \Omega$  ends in  $i^*$ , so we can define  $\lambda : \Omega \rightarrow \delta_R^-(i^*)$  such that  $\lambda(p)$  is the last arc of  $p$ . Thus, we can figure out that paths and ratings are coupled, as follows:

$$\rho(p) = \rho(\lambda(p)).$$

Let  $\Theta = \{\sigma \in [-1, 1] \mid \exists p \in \Omega (\sigma = \rho(p))\}$  be the set of ratings for  $i^*$  available to provide a prediction for the rating of  $k^*$  about  $i^*$ . We evaluate each rating by assigning to it the sum of the confidences along the corresponding paths. Let  $v : \Theta \rightarrow \mathbb{R}_+$  be given by

$$\forall \sigma \in \Theta \quad v(\sigma) = \sum_{\substack{p \in \Omega \\ \rho(p) = \sigma}} \gamma(p).$$

We use  $v$  to define a ranking on  $\Theta$  (i.e. an order  $<$  on  $\Theta$ ): for all  $\sigma, \tau \in \Theta$  ( $\sigma < \tau \leftrightarrow v(\sigma) < v(\tau)$ ). Finally, the recommender system picks the most trustworthy  $\sigma$  in  $\Theta$  (i.e. the rating with highest associated cumulative confidence) as the prediction of the rating of the user  $k^*$  concerning the item  $i^*$ .

### 3.4 Recommender system 2: BMC

In this section we present a recommender system, called BMC, which is based on clustering techniques. It exploits only users' ratings, thus it can be classified as a collaborative filtering oriented system. Clustering is intensively used in the RS field. In fact, it is useful to cluster items or users into homogeneous groups because similar customers choose (presumably) similar objects, thus items chosen in the past give suggestions of new items to recommend. However it is not easy to find useful clusters because data are usually sparse and each user has considered, purchased or rated only a small portion of the whole set of possible items. In literature, examples and discussions of recommender systems which exploit clustering methods can be found, among others, in [176,154,41]. Clustering algorithms group users maximizing both intra-cluster similarity and dissimilarity between different clusters. They work essentially as follows.

#### General clustering based recommendation algorithm.

1. Calculate similarities between users<sup>4</sup>.
2. Produce  $c$  clusters of users (note that, depending on the clustering algorithm,  $c$  could be fixed *a priori* or not). Namely, the subset of all users  $U$  is partitioned in  $c$  mutually disjoint clusters  $U_1, U_2, \dots, U_c$  so that  $\bigcup_{i=1}^c U_i = U$ . As usual, users belonging to the same cluster are called neighbors (for simplicity, we use  $U_a$  to refer to the set of neighbors of  $a$ ).
3. Given an active user  $a \in U_a$  who needs a recommendation, use its neighbors' ratings to predict a suggestion, by means of a *prediction function* (often, an average):

$$\text{prediction}(a, j) = \frac{\sum_{u \in U_a} r_{u,j} \cdot \text{sim}_{a,u}}{\sum_{u \in U_a} |\text{sim}_{a,u}|} \quad (3.2)$$

where  $j$  is the item for which the active user needs a prediction,  $r_{u,j}$  is the rating of the neighbor  $u$  for item  $j$ , and  $\text{sim}_{a,u}$  is used to identify the similarity between the active user  $a$  and its neighbor  $u$ .

Step 2 of this process can be computationally difficult for large instances, so we have to look for specificities of the data in order to adopt convenient clustering strategies. A remarkable point is that data feeding pure collaborative filtering oriented recommender systems have a typical structure that make them suitable for a representation by means of bipartite networks. In fact there are no direct relationships between items or between users, but only connections between item and user.

---

<sup>4</sup>As mentioned previously, there are several techniques for calculating similarities between users (see [82] for details)



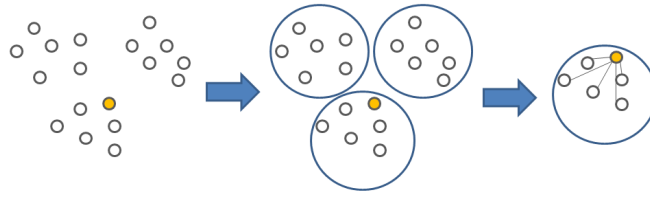


Figure 3.6: Clusters formation.

### 3.4.1 Bipartite networks and modularity

Formally, a bipartite network is a graph  $G = (V_1, V_2, E)$  which has two subsets of nodes  $V_1 = \{1, \dots, p\}$  and  $V_2 = \{p + 1, \dots, n\}$  and a set of edges  $E = \{1, \dots, m\}$  connecting pairs of nodes in different subsets. Bipartite graphs can represent properly data arising in the domain of recommender systems where there are users who give ratings to items. More precisely, one can associate the set  $V_1$  to the users and the set  $V_2$  to the items. An edge connects a user to an item for which that user has given a rating (which can be represented as the weight of the edge).

In this work (in order to boost clustering based collaborative filtering) we focus on clustering methods that fit bipartite networks. In particular we consider *modularity based* techniques that have been introduced by Newman and Girvan (and that, at the best of our knowledge, have never been used in this context) [128]. The idea underlying modularity based approaches is that the most relevant way to evaluate the links between groups in a network is a comparison between their effective quantity and the expected one, if the network would have been randomly generated. Basically, it is the “*statistically surprising arrangement of edges*” that gives us information, because “*... if the number within groups is significantly more, then it is reasonable to conclude that something interesting is going on*” [127]. We can precise and quantify this idea by means of a measure that is just called “*modularity*” and that is the fraction of edges within communities minus the expected fraction of such edges [128]. In case of bipartite unweighted undirected graphs, as reported by Barber [10] and Newman and Leicht [103] the modularity is defined as

$$Q_b = \frac{1}{m} \sum_{i=1}^p \sum_{j=p+1}^n \left[ \tilde{A}_{i,j} - \frac{k_i k_j}{m} \right] \delta(g_i, g_j), \quad (3.3)$$

where the adjacency matrix  $\tilde{A}_{i,j}$  is equal to 1 if there is an edge between nodes  $i$  and  $j$ , and 0 otherwise,  $k_i$  and  $k_j$  are respectively the degrees of nodes  $i$  and  $j$ , that is the number of edges incident with  $i$  and with  $j$ . Finally,  $g_i$  and  $g_j$  are the communities to which belong nodes  $i$  and  $j$ , and  $\delta$  is the Kronecker symbol, equal to 1 if  $g_i$  and  $g_j$  are the same, and 0 otherwise.

A good partition into communities should provide a high value of modularity.

Unhappily, the problem of modularity maximization is NP-complete [23]. For bipartite graphs the complexity is an open problem. However, the problem is not easy to solve, and some heuristics are proposed; for a review, see [62]. An interesting method is that of Liu and Murata called LPAb+ [112]. This heuristic combines a modified label propagation algorithm (LPAb') and a multistep greedy agglomerative algorithm to avoid local maxima.

### 3.4.2 RS by means of bipartite modularity based clustering

Our recommendation process is similar to the general clustering based recommendation one described in Section 3.4. However, there are three remarkable features that should be pointed out:

- clustering is based on bipartite modularity maximization.
- we reduce the graph to an unweighted one. We consider only the top ratings: there exists an edge between a user and an item if the user likes that item and the rating is maximum in the ratings range;
- we drop the prediction function and use directly clusters to provide suggestions.

Thus, our procedure reduces to the following two steps, as depicted in Fig. 3.7.

#### Modularity based recommendation algorithm:

1. Produce  $c$  clusters of the bipartite users-items graph with high bipartite modularity value ( $c$  is decided by the algorithm during computation).
2. If an active user  $a$  and an item  $j$  are in the same cluster and there is not an edge between them, then predict that the rating of  $j$  given by  $a$  would be the maximum in the rating range.

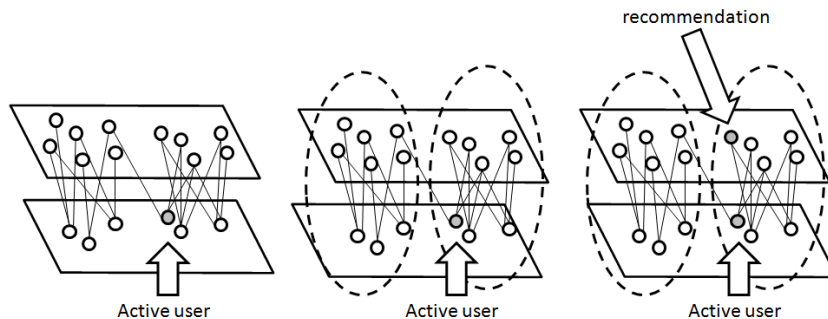


Figure 3.7: Modularity based recommendation.

Basically, we consider only maximum ratings to feed the recommender system and provide predictions that have a value that corresponds to the maximum possible.

### 3.5 Recommender system 3: LSPR

In this section we present another collaborative filtering oriented recommender system, called LSPR, that exploits Information Retrieval (IR) techniques (see [116] for an introduction to IR). In fact, as mentioned above, it was remarked (for example see [12]) that RS shares fundamental aspects with IR and there is somehow a continuity between these two fields of research. Thus, we have developed a recommender system that makes use of concepts and tools used elsewhere in an IR context, believing that the underlying structure could also provide an interesting framework for RS algorithms. IR methods deal with documents and terms. A common task is retrieving a set of documents in answer to a query. Thus, our idea is that methods that can find similar documents can be used to find similar users, because the problems are formally comparable, introducing the match we expose below.

The elements that compose a IR scenario are documents, terms and a query. First of all, notice that in IR a document is characterized by a set of terms. The presence of specific terms in a document and their frequency (their weights), define it. In RS a user is characterized by the set of items (that he/she has rated) and by ratings. We consider each user as a document, each item as a term and ratings as term weights. Moreover, we observe that the active user plays the same role played

IR	RS
document	user
term	item
query	active user
weight	rating

Table 3.1: Translation of IR elements into RS ones

by the query in IR contexts (see table 3.1). The meaning of this analogy is that in IR we want the documents more similar to the query, and for the RS problem we want the users more similar to the active user.

Once we have set this correspondence, we can use one of the several existing IR algorithms to obtain the list (called *ranking list* in IR) that represents the set of users more similar to the active user, ordered by decreasing similarity.

We can use this set of users to get the prediction for the active user.

#### 3.5.1 Model

This section describes the IR method that we adopt and exploit. Basically in the LSPR model [43] the query is viewed as a spectrum and each document as a set of filters, with one filter for each document term. Usually in IR two weighting schemes are necessary: the first one for the terms of the documents and the second one for the terms in the query. LSPR uses the TF-IDF weighting schema for the former, and the IDF weighting schema for the latter. In order to use the IR algorithms for

the recommender systems, it is necessary to modify these weighting functions. Since each user becomes a document, and each item becomes a term, there is a similarity between the matrix  $D$  and the well-known term-document matrix. At this point, consider an active user  $k \in U$ , for which we want to predict the rating for the item  $h \in I$ . Starting from  $D$ , we compute a new matrix  $WU$  (that plays the role of the normalized TF-IDF weights matrix in IR) as follows:

$$WU_{ij} = \begin{cases} 0 & \text{if } D_{ij} \cdot D_{ik} = 0 \\ 1 - \frac{|D_{ij} - D_{ik}|}{4} & \text{otherwise.} \end{cases} \quad (3.4)$$

This means that the more the rating of a user for a item is similar to the rating of the active user, the more its weight (from 0 to 1).

The column  $k$  in this matrix is not considered, because it is 0 for the items not rated by the active user, 1 otherwise.

After that, we compute the weights for the active user (i.e. the IDF weights for the query); we save these informations in the column  $k$  of the matrix  $WU$ , using the following formula:

$$WU_{ik} = \begin{cases} 0 & \text{if } n_i = 0 \text{ OR } D_{ik} = 0 \\ \log_2 \left( \frac{n}{n_i} \right) & \text{otherwise,} \end{cases} \quad (3.5)$$

where  $n_i$  is the number of users that have rated the item  $i \in I$ , and  $n$  is the total number of users.

Now we are ready to use the IR algorithm: the query is represented by the column  $k$  of  $WU$ , while the documents of the collection are the columns  $j \neq k$  of the same matrix with  $WU_{hj} \neq 0$ . The output of the model is the ranking list of the documents, ordered by increasing relevance. This means that the collection is the set of users that have rated the item  $h$ , and the output is the same set of users ordered from the more to the less “similar” to the active user.

The last operation is to predict the rating. To do this, we use the ratings of the users in the ranking list, weighted by their rank, so that the smaller the rank of the user is, the more his rating is considered. Suppose the ranking is given by the list of users  $R$ , where  $|R|$  is the number of retrieved users, the rank of each user is from 0 (first) to  $|R| - 1$  (last), and  $D_{h,j(r)}$  is the rating for the item  $h$  of the user with rank  $r$ . The predicted rating is computed as:

$$p_{hk} = \frac{\sum_{r=0}^{|R|-1} \left(1 - \frac{r}{|R|}\right) \cdot D_{h,j(r)}}{\lambda}, \quad (3.6)$$

where  $\lambda$  is the normalization term, computed as

$$\lambda = \sum_{r=0}^{|R|-1} \left(1 - \frac{r}{|R|}\right) = \frac{|R|+1}{2}. \quad (3.7)$$

Figure 3.8 summarizes the algorithm. Basically, the rows from 1 to 14 represent the operation described by equation (3.4), while the rows from 15 to 23 implement the equation (3.5). Finally there is the call to the IR algorithm, and the prediction of the rating, according to equations (3.6) and (3.7).

**Algorithm:** RecSys-to-IR

**Input:** data set  $D$ , active user  $k$ , item  $h$ , IR algorithm

**Output:** prediction  $p_{hk}$

```

1  for each  $i \in I$ 
2      do
3           $n_i \leftarrow 0$ 
4          for each  $j \in U | j \neq k$ 
5              do
6                  if  $(D_{ij} \cdot D_{ik} = 0)$ 
7                      then
8                           $WU_{ij} \leftarrow 0$ 
9                      else
10                          $WU_{ij} \leftarrow 1 - \frac{|D_{ij} - D_{ik}|}{4}$ 
11                          $n_i \leftarrow n_i + 1$ 
12                     end if
13             end for
14     end for
15     for each  $i \in I$ 
16         do
17             if  $(n_i = 0)$ 
18                 then
19                      $WU_{ik} \leftarrow 0$ 
20                 else
21                      $WU_{ik} \leftarrow \log\left(\frac{n_i}{n_i}\right)$ 
22                 end if
23     end for
24     Call the IR algorithm, and get the ranking list  $R$ 
25      $p_{hk} \leftarrow \text{Round} \left( 2 \cdot \frac{\sum_{r=0}^{|R|-1} \left(1 - \frac{r}{|R|}\right) \cdot D_{h,j(r)}}{|R|+1} \right)$ 
26     return  $p_{hk}$ 

```

Figure 3.8: The prediction algorithm.

## 3.6 Evaluating the recommender systems

Several measures have been proposed in order to evaluate the effectiveness of a recommender system. First of all, we use a metric that is popular in the Recommender Systems community because it was used during the Netflix Prize (an open competition for the best recommender system that was promoted by the American corporation Netflix and which is a milestone in the field of Collaborative Filtering [166]). It is the *accuracy* computed as the square root of the averaged squared difference between each prediction and the actual rating (the root mean squared error or “RMSE”), as we detail below. We believe that it represents a good choice since it is widespread and ensures somehow a standard benchmark. Nevertheless other metrics can be exploited to have a better evaluation (see [118, 82] for a discussion on this topic). In the next subsections we describe each one of the metrics we use.

### 3.6.1 Dataset

We use a standard dataset provided by GroupLens (one the most important research team in the RS community). It contains 100 000 ratings (from 1 to 5) given by  $n = 943$  users on  $m = 1682$  movies and each user rated at least 20 movies. The evaluation is performed using the “leave-n-out” approach [28]. The dataset is split into two parts. The ratings of the first part, called *test set*, are hidden. The ratings of the second part, called *training set*, are available to the recommender systems to learn users’ profiles, calculate similarities and/or clusters. The recommender system tries to predict properly the withheld ratings and we can verify if it works fine because we know the true values. We used five pairs (training set, test set) that share the same composition (80%/20% splits of the original data into training and test sets) as suggested by the guidelines of GroupLens itself. Notice that with this dataset, which contains only ratings and no direct trust relationships among users, TMW performs as a pure collaborative filtering oriented system (see Section 3.8 for a test with a different dataset that includes social relationships).

### 3.6.2 Evaluating effectiveness

#### 3.6.2.1 Accuracy

Let  $\bar{U}$  be the set of active users,  $\bar{I}_k$  be the set of movies rated by an active user  $k \in \bar{U}$ ; let  $p_{ik}$  denote the predictions generated by a certain algorithm for the movie  $i$  and the active user  $k$ , while  $r_{ik}$  is the corresponding real rating. RMSE is defined by

$$\text{RMSE} = \sqrt{\frac{\sum_{k \in \bar{U}} \sum_{i \in \bar{I}_k} (r_{ik} - p_{ik})^2}{\sum_{k \in \bar{U}} \sum_{i \in \bar{I}_k} 1}}. \quad (3.8)$$

We calculate the predictions with the systems using data from the training set

and we compare the prediction against the real rating in the test set. Moreover, (as explained in the example below) we employ the *community average* for a certain item (that is the average of the ratings given by all the users who vote the item) as a benchmark, with the aim of measuring how much our algorithm can improve the simple *community recommendation*. Thus, we also compute the RMSE of the community recommendation with respect to the actual ratings provided by the users.

Consider the following example, in order to clarify the use of RMSE and *community average*. We have in our dataset 5 items, and 5 users who rate some of these items (from 1 to 5, 0 means no rating); table 3.2 shows these ratings. There are 2

item\user	1	2	3	4	5
1	0	3	0	0	2
2	0	5	4	0	2
3	2	1	0	3	3
4	1	3	3	2	0
5	0	2	2	4	3

Table 3.2: Ratings

active users for whom we want to predict some ratings, as shown in table 3.3. The

item\active users	a	b
1	0	1
2	5	3
3	0	0
4	2	0
5	0	3

Table 3.3: Active users

aim of each recommender system algorithm is to predict the ratings of the active users; in our example, the ratings of the user  $a$  for the items 2 and 4, and the ratings of the user  $b$  for the items 1, 2 and 5, trying to find values closer to the real ones reported in table 3.3. According to the notation used in equation (3.8), we have  $\bar{U} = \{a, b\}$ ,  $\bar{I}_a = \{2, 4\}$ ,  $\bar{I}_b = \{1, 2, 5\}$ . Moreover,  $r_{a2} = 5$ ,  $r_{a4} = 2$ ,  $r_{b1} = 1$ ,  $r_{b2} = 3$ ,  $r_{b5} = 3$ . To show what is the *community average*, consider  $p_{a2}$ : it is the average of the ratings of the users who vote the item 2 (rounded to the closest integer, since the ratings are integer). This means that

$$p_{a2} = \text{Round} \left( \frac{5 + 4 + 2}{3} \right) = 4.$$

In a similar way, we obtain  $p_{a4} = 2$ ,  $p_{b1} = 3$ ,  $p_{b2} = 4$ ,  $p_{b5} = 3$ . Now we have all the elements to compute the RMSE, and we obtain the following result

$$\text{RMSE} = \sqrt{\frac{6}{5}} = 1.095.$$

For our experiments we proceed as follows: we calculate the predictions with all the three algorithms and store all prediction results. Basically, for every rating we record a line in a log file with the predictions about that item and user of each one of the recommender systems and the community average, respectively, as shown in Table 3.4.

rating (real)	Rec. Sys. 1	Rec. Sys. 2	Rec. Sys. 3	community average
$R(u_i, p_k)$	$Pr_{RS_1}$	$Pr_{RS_2}$	$Pr_{RS_3}$	$Pr_{comm.aver.}$

Table 3.4: Log line for accuracy calculation.

Table 3.5 shows the outcome. Five rows correspond to five different subdivisions in training and test sets, as mentioned above.

set	TMW	BMC	LSPR	community
1	0.956	1.025	0.985	1.073
2	0.948	0.998	0.974	1.067
3	0.953	1.010	0.971	1.060
4	0.951	1.080	0.967	1.056
5	0.952	1.045	0.975	1.065

Table 3.5: RMSE

In Table 3.6 we express this result in relative terms by providing the *rate of improvement* with respect to the average of the ratings by the community.

set	TMW	BMC	LSPR
1	10.9 %	4.5 %	8.2 %
2	11.2 %	6.5 %	8.7 %
3	10.1 %	4.7 %	8.4 %
4	9.9 %	-2.3 %	8.4 %
5	10.6 %	1.9 %	8.5 %
mean	10.5 %	3.1 %	8.4 %

Table 3.6: Improvement over community average.

### 3.6.2.2 Audacity

We would like to provide users with the maximum possible number of good recommendations. Moreover, one of the sub-goal of the system is to be interesting. This means that we prefer a recommender system that tries to suggest a lot of “five stars” items to one that recommends only common and known items. We could say that we prefer a “audacious” system rather than a too “timid” one. If a recommender never proposes a top rating prediction and it always predicts a rating that is average, it might result quite accurate, but, definitely, it does not provide a good service. Users look for great restaurants, movies and books, not for unimpressive ones. Thus, we



introduce another measure. We define *audacity* the portion of predictions that attain the maximum possible rating in the rating scale. A high audacity means that the recommender system tries to be unequivocal and provide sharp judgments. Thus we count the portion of top ratings that the system propose. Table 3.7 reports the results

set	TMW	BMC	LSPR
1	18.5%	100%	19%
2	18.8%	100%	17.3%
3	21.3%	100%	18.4%
4	19.2%	100%	18.2%
5	19.1%	100%	19.0%
mean	19.4%	100%	18.4%

Table 3.7: Portion of top rating predictions

Notice that BMC is extremely audacious. This depends on the fact that it builds clusters considering only the arcs that correspond to top ratings, thus it is deliberately designed to recommend only top rated items.

### 3.6.3 Evaluating efficiency

A widespread distinction, in RS community, is between systems that shall work *online* (i.e. on-the-fly) and system that can work *off-line*. In the first case, the system shall be able to recalculate very quickly its predictions each time a user rates an item. In the second case, the system can provide its recommendation asynchronously (for example through periodical newsletters) and does not have to be very fast. We observe that computational efficiency is one of the objective of RS design, but not the most important one. RS are realized to achieve users' satisfaction. Fast systems that provide poor recommendations do not get the goal.

Nevertheless, in order to evaluate efficiency, we record the time the systems need to produce their recommendations. Table 3.8 shows the outcome of our tests, for each pair training/test set. Notice that we require the predictions for all the hidden ratings of the test set, for all users. This is a big job. In real online scenario, normally, the system is required to provide a few predictions for one single user at a time. Tests are performed on a AMD Sempron Mobile, 2 Ghz, 3 Gb ram. Algorithms are implemented in java.

Even from this point of view, BMC behaves quite differently from the other systems. It is remarkably faster. Nevertheless, we observe again that it considers only the top rated items, namely a subset of the training set. This reduces considerably the computational effort, but entails the important drawback of a poor accuracy.

set	TMW	BMC	LSPR
1	824	591	2531
2	1023	623	3753
3	1265	685	4176
4	1284	698	4350
5	1367	688	4397
mean	1095	657	3841

Table 3.8: Computation time (seconds)

### 3.7 Recommender systems design problem

We have presented three systems that perform differently from several points of view, hence, the choice of the best system is not straightforward. In this section, we analyze the problem of choosing among a set of available recommender systems.

When we design a recommender system there are conflicting aims. We have selected three goals and would like to have an accurate, audacious and fast system. From the point of view of the final decision, each system represents a possible choice that is characterized by its accuracy, audacity and computational efficiency. This defines a multicriteria problem that has a finite set of explicitly known choices. This corresponds to a typical scenario 1 of MCDM (see Section 1.2.1). We want to rank the three systems exploiting the tests we have done. However the metrics we have used are not dimensionally homogeneous, thus, quite classically, we need a method to make comparisons possible. We adopt very simple functions. Each function maps a system into an *absolute rating* that is equal to the ranking, in reverse order, of the recommender system in a specific test.

Rank	Utility
1	3
2	2
3	1

Table 3.9: Mapping functions. If a recommender system is the best in a test then its utility is 3.

The range of the functions is the interval  $Rank = \{x : 1 \leq x \leq 3, x \in \mathbb{N}\}$ . For example, if a system was ranked first in the efficiency test, its score with reference to this attribute is 3, as showed in table 3.9. More formally, let  $S_{rs} = \{TMW, LSPR, BMC\}$ . We introduce three functions that map the performance in a test into an element of  $Rank$  (i.e. a numeric value).

- $U_{cpu} : S_{rs} \rightarrow Rank$
- $U_{acc} : S_{rs} \rightarrow Rank$
- $U_{aud} : S_{rs} \rightarrow Rank$

Each system is now associated to a triplet that qualifies its global performance. For example, a triplet such as (3, 3, 3) would mean that the system performs better than the others from all points of view. We want to solve the following problem.

$$\left. \begin{array}{l} \max_s U_{cpu}(s) \\ \max_s U_{acc}(s) \\ \max_s U_{aud}(s) \\ \text{s.t. } s \in S_{rs}, \end{array} \right\} \quad (3.9)$$

The problem has three criteria. If we assume, for simplicity, that the criteria have equivalent importance, we obtain the Table 3.10 that shows that the LSPR is the worst system. TMW and BMC get the same score.

	cputime	accuracy	audacity	final rating
TWW	2	3	2	7
BMC	3	1	3	7
LSPR	1	2	1	4

Table 3.10: Evaluation matrix.

If we privilege, among objectives, computational efficiency or audacity, BMC results even better. It would be the best system. Nevertheless, this would be quite misleading. BMC provides quickly hazardous recommendations with a poor accuracy. It behaves as a friend who suggests you, quickly, enthusiastically and indifferently, bad and good restaurants with no much awareness. This is not the kind of recommender that people normally look for. In our opinion, BMC is a good option only when we are short of time. If we privilege accuracy, then TMW is the best system. We think it is a better choice. Clearly, this brief analysis shows the fundamental importance of a good balance of the criteria in order to take the final choice. The balance depends on the preferences of the decision-maker and on the capacity of gathering them (this is a huge topic and is out of the scope of this work, see [132]).

### 3.8 Evaluation TMW using a small dataset

TMW has been used as engine of a working website. The system was developed for “DisMoiOu” (“TellMeWhere”) an on-line service<sup>5</sup>, that provides the user with advice on places that can be interesting for him/her. This gave us the possibility of applying a different testing method, due to the recording of ratings and predictions live. Unhappily, due to the policy of the enterprise, we could not use this method to test BMC and LSPR.

With live data, we don’t need to simulate the prediction process because we can compare predictions with real ratings from users *on-the-fly*. For our experiment we

<sup>5</sup><http://dismoiou.fr>, <http://tellmewhere.com/>

proceed as follows: every time a user provides a rating, we calculate the predictions with TMW using the entire dataset (with the exception of the rating entered), and store all the prediction results. Basically, for every rating, we record a line in a log file with the predictions of TMW, about that item and user. As a benchmark to evaluate the algorithm we employ the *community average*, as usual. The result reported in the following table refers to the analysis we performed using the ratings we got over the first four months of activity of the website. We had 315,463 ratings (from 1 to 5), inserted by 69,794 users on 147,319 items (places). Table 3.11 reports the overall RMSE computed in that period.

comm.	TMW
1.0710	0.9865

Table 3.11: RMSE for all ratings

TMW improves community ratings by only 7.89% on average. Notice that in this situation TMW can also exploit explicit friendships statements among users, which are included in the database. Despite to this advantage, TMW performs worse in this case than in the previous tests. We suspect that the results obtained are due to the structure and the dimension of the dataset. Some known RS dataset are bigger. Netflix consists of 100 million ratings provided by over 480 thousand users, on nearly 18 thousand movie titles. Jester Joke dataset has 4.1 million by 73,496 users on 100 jokes. The dataset of GroupLens that we have used is smaller (since it has 100,000 ratings) but each user rated at least 20 movies. On the contrary, the dataset of DisMoiOu is sparse. Most users have expressed only a few ratings. The main issue, which arises from this experience, is that it is not clear which is the minimal dimension of a dataset to make it a reliable base to build a test bed. This a very important question in our opinion and our impression is that it has been underestimated. Of course, if a dataset is untrustworthy then tests can be misleading. This experience has shown that from the practitioner's point of view, tuning a good recommender system relies on finding a reliable dataset and test bed, and that this issue has not been addressed adequately in the RS literature.



# The equitable hazmat transportation problem

## 4.1 Introduction

The term “sustainable development” refers to the problem of balancing economic growth and ecological issues. This theme has been attracting great consideration in the last decades, due to an increasing awareness of the fragility of the natural environment. MCDM methods appear to be an effective framework to deal with the conflicting needs that characterize environment related scenarios. In fact, there is a stable tradition of application of MCDM techniques to ecological issues [59, 122].

In this chapter, we consider the problem of hazardous materials transportation on a road network (Hazmat transportation problem), that is relevant from an ecological point of view, due to the environmental consequences of possible accidents.

This kind of transportation can involve one origin-destination pairs, in which case it remains a *local* routing problem. Nevertheless, in general, it concerns many origins and destinations, yielding a *global* routing problem [111]. We can figure it this way: there are  $N$  trucks that have to transport some kind of dangerous material from one or many production points to one or many garbage dumps, crossing different areas, and we have to select a set of paths that is optimal from the point of view of cost, risk and equity. However, equity is somehow unusual and is hard to define. We consider and compare two different ideas of equity. The first approach simply requires that all areas involved in the transportation network share the same level of risk, namely that the difference of the risk of two areas is inferior to a fixed threshold. This is a fair and intuitive idea, but it could also lead to improper solutions where risk is similar but uniformly high. The second definition of equity we use is inspired by the concept of “*equity as fairness*” of J. Rawls [143, 144, 145]. It is an application of his *difference principle* that allows disparities in the distribution of goods only if those disparities benefit the poorer members of society (see Section 5.5 for a discussion).

Basically, in this context, the difference principle means that we may introduce disparities only if they advantage the worst-off zone, namely reduce the risk of the less favorite area (the most exposed to the risk). We look for a rationale for choosing a particular definition of equity (for hazmat transportation), thus we investigate the relationships between each definition of equity and the consequences it entails.

## 4.2 Related works

Many authors suggest that the need for equity in a transportation problem can be achieved finding a set of different paths, the more different, the better. In fact, if paths are different, they cross different zones and they spread the risk over a big area. This approach reduces our problem to the Path Dissimilarity Problem (PDP). Most of the methods for PDP have two phases: in the first phase a set of paths that fulfil a requirement (typically that minimize length or cost) is identified, in the second phase the most dissimilar paths are selected among them. In this case equity is ensured *a posteriori*. Equity itself is not explicitly defined and it is assumed that the dissimilarity among paths assures it.

We briefly recap some of the most relevant works in this field.

- Iterative Penalty Method, Johnson et al. [86], 1992.

This method includes two phases. The authors consider a single one origin-destination pairs, thus, they initially look for the shortest path. Subsequently, weights on the arcs of the selected path are increased in order to penalize it. Then they look again for the shortest path, according to the new weights. The process is repeated  $k$  times, to obtain  $k$  different paths.

- Gateway Shortest Path (GSP) method, Lombard and Church [113], 1993.

This method includes two phases. In the first, the authors generate a set of different paths by forcing them to go through different selected nodes (gateways). In the second phase dissimilar ones are selected by means of a dissimilarity measure between paths, that is defined as the absolute difference between the “area-under-the-path” of two paths (i.e. the area between a path and an axis).

- Minimax method, Kuby et al. [96], 1997.

This method includes two phases. In the first phase a set of paths is obtained solving the  $k$ -shortest paths problem. In the second phase a subset of short, dissimilar paths is constructed step by step, exploiting a criterion that is a linear combination of length and similarity. The similarity  $d^s(P_1, P_2)$  of two paths  $P_1$  and  $P_2$  is calculated as the length of their shared portion. The first path  $P_1$  that is selected is always the shortest path; the second path is selected in order to minimize its length and minimize the similarity with  $P_1$ . The third

path  $P_3$  is selected in order to minimize the length and minimize the similarity with  $P_1$  and  $P_2$ . The procedure is repeated until the desired number of paths is obtained.

- Akgun et al. [4], 2000.

This method includes two phases. In the first phase the authors build a set of candidate paths, either solving a  $k$ -shortest paths problem or applying the iterative penalty method. Then, in order to build a subset of dissimilar paths, they solve a discrete  $p$ -dispersion subproblem ( $p$  out of  $m$  given elements ( $p < m$ ) are selected to maximize the minimum distance between any two of the selected elements). A similarity measure is defined as the average ratio between the length of the shared portions of two paths and the length of the whole path. In particular, the authors define the similarity  $S(P_i, P_j)$  between two paths  $P_1$  and  $P_2$  as  $\frac{1}{2} \left( \frac{L(P_1 \cap P_2)}{L(P_1)} + \frac{L(P_1 \cap P_2)}{L(P_2)} \right)$ , where  $L(P_i)$  denotes the length of path  $P_i$  and  $L(P_n \cap P_m)$  denotes the length of the shared portions of path  $P_n$  and path  $P_m$ . The dissimilarity  $D(P_i, P_j)$  of two paths  $P_i$  and  $P_j$ , is  $D(P_i, P_j) = 1 - S(P_i, P_j)$ . Using this dissimilarity as a distance they can reduce the dissimilarity problem to the discrete  $p$ -dispersion one.

- Dell’Olmo et al. [48], 2005.

Similarly to other methods, this method includes two phases. Dell’Olmo et al. stress the drawbacks of the approaches proposed by previous works. In the context of the hazmat problem, there are at least two objectives. In fact, normally, minimal distance and minimal risk are both required. They observe that the construction of the candidate set is realized using only one criterion or a linear combination of two objectives (a technique that implies an “a priori” choice of the relative importance of the objectives). Thus, in the first phase, they generate the whole set of Pareto optimal solutions by solving a Bi-criteria Shortest Path Problem. In the second phase they look for the most dissimilar paths in the candidate set that contains all the efficient solutions. The authors introduce the notion of “buffer zone” that is the area near a path that could be impacted by an accident. The similarity between two paths corresponds to their shared buffer zone. This similarity is used to choose the most dissimilar path by means of a reduction to the discrete  $p$ -dispersion problem.

- Carotenuto et al. [36], 2007.

The authors distinguish between the risk (for the population due to hazmat transportation) on a path and on an edge of the that path. They require that the risk of the zone near an edge (called *buffer zone*) remains below a specific threshold. Basically, this constraint prevents from choosing always the same edges (if an edge is chosen too often, the region near the edge has a considerable



risk). In this method, the two phases are not clearly distinguished. Moreover, no explicit definition of dissimilarity between paths is used.

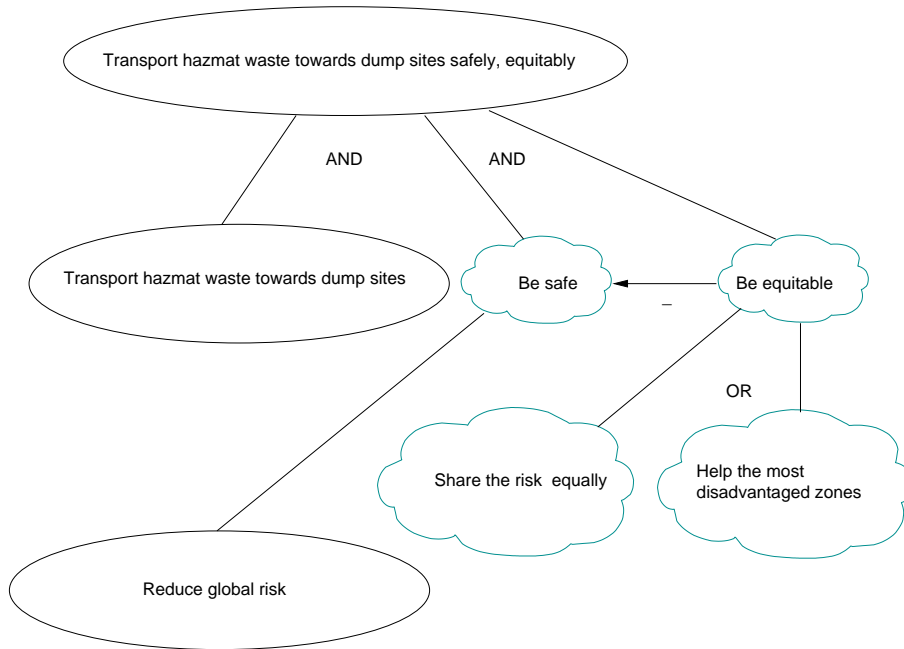


Figure 4.1: Goal Diagram

### 4.3 Operational model of a transportation system

Our target system is the whole *transportation system* that includes road network, hazmat sources, waste disposal plants. Tables 4.1-4.4, which are based on [35], shows some relevant stakeholders of our target system, that can affect or be affected by the transportation of hazardous material. Among others, waste producing industries, national government agencies and political jurisdictions, local governments, transportation industry, the media, public sector interest groups or trade associations, public health and environmental interest groups.

A detailed model of the scenario we are considering would involve many details. Nevertheless, as we have explained in the Chapter 1, a top-down approach goes through the definition of several models with different levels of abstraction. We focus on the system-level perspective, namely we aim to provide a prototype, overriding some stakeholders and specific technical features (of the lorries, of the roads, of the transported materials etc.).

#### 4.3.1 Global Goal

The high-level requirement, which sets the general goal we deal with, is:

institution	Generic concern(s)	con-	Specific concern(s)	con-	Need(s)
PUBLIC SECTOR - FEDERAL					
Shipper	Provide for environmentally sound transport and disposal of hazardous materials	en-	Assure that hazmat transportation complies with planned generation, processing and disposal schedules	con-	Maintenance of planned hazardous materials program and schedule
Traffic managers	Arrange for, negotiate and manage HMT	ne-	Negotiate shipping rates, schedules, operating conditions with carries; ensure that carriers meet laws and regulations	con-	Maintain sufficient authority to ensure timely transport of such materials
Department of transportation	Hazardous materials be transported in a manner that is safe and poses minimum threat to public safety	ma-	Compliance of affected parties with statutes and regulations	con-	Coordination with relevant federal and state authorities

Table 4.1: Environment of the hazmat transportation problem (a)

institution	Generic concern(s)	con-	Specific concern(s)	con-	Need(s)
PUBLIC SECTOR - STATE					
Governor	Protect interests of his/her electorate	elec-	Protect public health and welfare	pub-	participation in hazmat transportation planning and implementation
Representatives of affected districts	Protect interests of district's citizens	con-	Equitable distribution of costs	con-	mitigation for potential adverse impacts of accidents
Department of transportation	Safe transport of hazardous materials	con-	select preferred routes for large shipments	con-	approval of transportation route and vehicles
Environmental Protection Agency	Environmentally sound transport	con-	shipper/carrier compliance	con-	monitoring of transport

Table 4.2: Environment of the hazmat transportation problem (b)

institution	Generic cern(s)	con-	Specific cern(s)	con-	Need(s)
PUBLIC SECTOR - LOCAL					
Major	Protect of electorate	interests	Equitable distribution of costs	distri-	participate in state's selection of preferred route
Fire Depart- ment	Protection of life		Adequacy of existing resources to participate to hazmat trans- portation scenar- ios		Provision of tech- nical and financial assistance

Table 4.3: Environment of the hazmat transportation problem (c)

institution	Generic cern(s)	con-	Specific cern(s)	con-	Need(s)
PRIVATE SECTOR					
Carrier	Safe transport of hazmat materials		Compliance with regulations and other require- ments		Ability to de- termine proper routes within constraints
Mass media	Ability to pro- vide accurate re- portage		Being provided comprehensive and accurate information		Access to relevant information
Environmental interest groups	Protection of public Health		Potential adverse impact to public health		monitoring of transport dis- posal

Table 4.4: Environment of the hazmat transportation problem (d)

- The transportation system shall ensure safe disposal of hazardous waste in such a way that the risk of potential catastrophic accident is equitable over the population.

Fig. 4.1 depicts the related goal diagram. Notice that, *be safe* and *be equitable* are non-behavioral requirements. Our working hypothesis is that safety and equity are contradictory goals.

In the following sections, we make all these requirements hard by integrating them in MP formulation and check this hypothesis, examining the related trade-off.

#### 4.4 Mathematical programming formulation

In this section we formalize our problem introducing a mathematical programming formulation. First of all, we observe that it is quite natural to represent our scenario, that we can figure out as a road network, by a means of graphs. Thus, we introduce

the related terminology. Secondly, we also remark that our problem, apart from the requirement of being equitable, shares several features with the Minimum Cost Multi-Commodity Flow Problem [58]. Therefore the following model is inspired by the formulation of this problem and we use the term *commodity* to denote an hazardous material that we need to dump.

- Let  $G = (V, A)$  be a directed graph, modeling a road network.
- For  $v \in V(G)$  let  $N^+(v) = \{u \in V \mid \{v, u\} \in A\}$  be the *forward star* of  $v$
- For  $v \in V(G)$  let  $N^-(v) = \{u \in V \mid \{u, v\} \in A\}$  be the *backward star* of  $v$

#### 4.4.1 Sets, variables, objectives, constraints

##### 1. Sets:

- We consider many commodities, which we identify by means of a set of unique keys; thus, we introduce the set  $K = \{1, \dots, K_{\max}\}$  of commodity indices.
- Typically we can imagine that the geographic area, covered by the road network, is divided into administrative zones; we introduce the set  $Z = \{1, \dots, Z_{\max}\}$  of zones.
- For the sake of simplicity we assume that each road (arc) belongs to a single zone. Hence we can associate each zone to a subset of the set of arcs, so that these subsets are disjointed, through the following function  $\zeta : Z \rightarrow \mathcal{P}(A)$ . Hence,  $\forall z \in Z$ , we indicate with  $\zeta_z \subseteq A$  the set of roads of the zone  $z$ .

##### 2. Parameters.

- Each arc  $(u, v)$  has a positive traversal cost and a capacity, thus we introduce the following functions that attribute arc weights and capacities:
  - (a)  $l : A \rightarrow \mathbb{R}_+$  (lengths, traveling time or traversal cost)
  - (b)  $C : A \rightarrow \mathbb{R}_+$  (arc capacity)
- Each commodity has an origin, a destination, a “demand”, i.e. the quantity of that commodity that has to be dumped. Moreover, the garbage dumps can contain a limited amount of a specific commodity. We introduce mappings between the set of commodities and the sets of its attributes as follows:
  - (a) Map  $s : K \rightarrow V$  (source nodes for each commodity)
  - (b) Map  $t : K \rightarrow V$  (target nodes for each commodity)
  - (c) Map  $d : K \rightarrow \mathbb{R}$  (quantity to dump for each commodity)

- (d) Map  $c : K \rightarrow \mathbb{R}$  (capacity of the garbage dumps for each commodity)
- We assume also that we know the probability of an accident that may occur on a road (arc), and that the potential damage depends on the kind of commodity involved in the accident, so that we can evaluate the risk as the product of these two aspects (probability and damage). We call it *traditional risk* (see [57, 56] for a discussion on the modeling of transportation risk). Therefore, basically, we assume that risk is a given input of the problem.
- (a) Map  $p : A \rightarrow [0, 1]$  (probability of accident on arc)
- (b)  $\Delta : A \times K \rightarrow \mathbb{R}_+$  (damage caused by accident, by unit of commodity on an arc)
- (c) For  $(u, v) \in A, k \in K : r_{uv}^k = p_{uv} \Delta_{uv}^k$  (traditional risk)

### 3. Decision variables.

- $x : A \times K \rightarrow \mathbb{R}_+$ : flow of commodity on arc

### 4. Objective functions.

There are several objectives that we can consider. We focus on total damage and equity.

- Objective function 1: total damage.

$$\min \sum_{\substack{(u,v) \in A \\ k \in K}} \Delta_{uv}^k x_{uv}^k \quad (4.1)$$

- Objective function 2: equity.

Our problem has a special aim. In fact, the goal concerning equity is the most peculiar ones of our scenario. Equity is not a easy concept. There are several ways to define it. We consider two possible formulations.

- (a) Risk sharing: the first one simply limits the difference in risk, in a pairwise comparison between zones (minimize pairwise risk difference):

$$\min \sum_{(z < w \in Z)} \left| \sum_{\substack{(u,v) \in \zeta_z \\ k \in K}} r_{uv}^k x_{uv}^k - \sum_{\substack{(u,v) \in \zeta_w \\ k \in K}} r_{uv}^k x_{uv}^k \right| \quad (4.2)$$

- (b) Rawls' principle: the second formulation exploits ideas proposed by J. Rawls who thinks that equity consists in aiding the most disadvantaged one (minimize the risk of riskiest zone):

$$\min \max_{z < w \in Z} \sum_{\substack{(u,v) \in \zeta_z \\ k \in K}} r_{uv}^k x_{uv}^k \quad (4.3)$$

## 5. Constraints.

Basic constraints:

- Arc capacity:

$$\forall (u, v) \in A \quad \sum_{k \in K} x_{uv}^k \leq C_{uv} \quad (4.4)$$

- Zero flow into sources:

$$\forall k \in K \quad \sum_{v \in N^-(s_k)} x_{vs_k}^k = 0 \quad (4.5)$$

- Zero flow out of targets:

$$\forall k \in K \quad \sum_{v \in N^+(t_k)} x_{t_k v}^k = 0 \quad (4.6)$$

- Flow conservation:

$$\forall k \in K, v \in V \setminus \{s_k, t_k\} \quad \sum_{u \in N^-(v)} x_{uv}^k = \sum_{u \in N^+(v)} x_{vu}^k \quad (4.7)$$

- Garbage dumps capacity:

$$\forall k \in K \quad \sum_{v \in N^-(t_k)} x_{vt_k}^k \leq c_k \quad (4.8)$$

- Demand (quantity of commodity that has to be dumped):

$$\forall k \in K \quad \sum_{v \in N^+(s_k)} x_{s_k v}^k = d_k \quad (4.9)$$

Notice that, if we consider only the Objective 4.1, the Constraints 4.4 -4.8 and we impose a strict equality to Constraint 4.8, then the problem reduces to a *minimum cost multi-commodity flow problem*.

### 4.4.2 Reformulation of the model

The formulation we have introduced so far, is multi-objective. To deal with this formulation, we need a MOMP method. We use the  $\epsilon$ -constraint method, to avoid possible problems due non-convexity of the Pareto frontier.

However, preliminarily, we remark that we can slightly change the MP formulation to ease it management. Procedures that transform a given formulation into another formulation are called “reformulations”.

“It is well known that several different formulations may share the same numerical properties (feasible region, optima) though some of them are easier to solve than others with respect to the most efficient available algorithms. . . . When a problem with a given formulation  $P$  is cast into a different formulation  $Q$ , we say that  $Q$  is a reformulation of  $P$ .” [106].

This is common technique in MP practice that has been recently systematized, in fact, according to [107], there are four important families of reformulations in MP: exact reformulations (which preserve all the optima of the original problem), narrowings (which preserve at least one optimum of the original problem), relaxations (which provide guaranteed bounds to the optimal objective function value of the original problem) and approximations (which are “asymptotic” exact reformulations: they are exact in the limiting value of some parameter). We use the first type of reformulation.

Focusing on the objective function 4.3, we notice that, in general, when we are given a problem in the form:

$$\min_x \max_{i \in I} f_i(x) \quad (4.10)$$

where  $i$  is a parameter which varies in the set  $I$ , we can reformulate it, introducing an additional variable  $\alpha$ :

$$\left. \begin{array}{l} \min_{\alpha} \quad \alpha \\ \text{s.t.} \quad \forall i \in I, f_i(x) \leq \alpha \end{array} \right\} \quad (4.11)$$

Hence, similarly, we can introduce in our case a new variable  $\alpha$  and reformulate the objective (4.3) in the following way:

$$\left. \begin{array}{l} \min_{\alpha} \quad \alpha \\ \text{s.t.} \quad \forall z \in Z, \Omega \leq \alpha \end{array} \right\} \quad (4.12)$$

where we have set:

$$\Omega = \left( \sum_{\substack{(u,v) \in \zeta_z \\ k \in K}} r_{uv}^k x_{uv}^k \right) \quad (4.13)$$

Then, we apply the  $\epsilon$ -constraint method.

#### 4.4.2.1 $\epsilon$ -constraint method

The basic idea of the method consists in the minimization (or maximization) of only one objective function  $f_{\epsilon}$  and the transformation of all the others in constraints of the following type,  $f_k(x) \leq \epsilon_k$ , (where  $\epsilon_k$  are convenient bounds) that reduce the search space. It has been introduced to overcome the main drawback of common scalarization methods (for example the *weighted sum method*) that fail to find solu-

tions if the objectives space is not convex . It is a variation the method developed by Haimes et al. [80]. Thus, the general MOMP:

$$\left. \begin{array}{l} \min_x F(x) = [f_1(x), f_2(x), \dots, f_{k^*}(x)]^T \\ \text{s.t. } \forall i \quad 1 \leq i \leq m \quad g_i(x) \geq 0, \\ \quad \forall j \quad 1 \leq j \leq n \quad h_j(x) = 0, \\ \quad x \in X, \end{array} \right\} \quad (4.14)$$

can be reformulated as follows:

$$\left. \begin{array}{l} \min_x f_\epsilon(x) \\ \text{s.t. } \forall k \quad k \neq \epsilon \quad f_k(x) \leq \epsilon_k \\ \quad \forall i \quad 1 \leq i \leq m \quad g_i(x) \geq 0, \\ \quad \forall j \quad 1 \leq j \leq n \quad h_j(x) = 0, \\ \quad x \in X, \end{array} \right\} \quad (4.15)$$

Varying  $\epsilon_k$ , alternative solutions are obtained. Optimal solution of (4.15) are weakly efficient for (4.14). If there is only one solution , then it is Pareto optimal, but usually uniqueness is hard to verify.

The application of the reformulation technique proposed above and of the  $\epsilon$ -constraint method, change our MP formulation of the problem. We transform the objectives 4.2 and 4.3 in the following two constraints.

## 6. Additional (reformulated) constraints.

- Risk sharing:

$$\forall z < w \in Z \quad \left| \sum_{\substack{(u,v) \in \zeta_z \\ k \in K}} r_{uv}^k x_{uv}^k - \sum_{\substack{(u,v) \in \zeta_w \\ k \in K}} r_{uv}^k x_{uv}^k \right| \leq R_D \quad (4.16)$$

where the scalar  $R_D$  is the threshold for the difference of risk. Through algebraic transformations we can dispose of the absolute value, and obtain the following two constraints.

$$\forall z < w \in Z \quad \left( \sum_{\substack{(u,v) \in \zeta_z \\ k \in K}} r_{uv}^k x_{uv}^k - \sum_{\substack{(u,v) \in \zeta_w \\ k \in K}} r_{uv}^k x_{uv}^k \right) \leq R_D \quad (4.17)$$

$$\forall z < w \in Z \quad - \left( \sum_{\substack{(u,v) \in \zeta_z \\ k \in K}} r_{uv}^k x_{uv}^k - \sum_{\substack{(u,v) \in \zeta_w \\ k \in K}} r_{uv}^k x_{uv}^k \right) \leq R_D \quad (4.18)$$



- Rawls' principle:

$$\forall z < w \in Z \quad \sum_{\substack{(u,v) \in \zeta_z \\ k \in K}} r_{uv}^k x_{uv}^k \leq R_P \quad (4.19)$$

where the scalar  $R_P$  is the threshold for riskiest zone.

We consider two MP formulations. One includes Constraints 4.17 and 4.18. The other one includes Constraint 4.19. These two formulations embody the two ideas of equity introduced above. We evaluate in next section how equity, in these two different formalizations, influences global risk.

## 4.5 Computational results

We perform a series of tests to establish if our models can be utilized with realistic instances of our problem. We observe CPU time in function of the instance size.

We use the AMPL modelling environment [63] and the CPLEX 12.2 solver [85] running with its default configuration on a single 2.4 GHz Intel Xeon CPU with 8GB RAM. We generate instances randomly, considering five parameters: the number  $n$  of the vertices of the network (cardinality of vertex set), the probability  $p$  that an arc exist (graph density), the maximum capacity  $C$  on arcs, the number  $K$  of commodities and the number  $Z$  of administrative zones (notice that in the following tables we use these abbreviations “Card.” for graph cardinality, “Gr.d.” for graph density, “Cap.” for arcs capacity, “Comm.” for number of commodities).

### 4.5.1 Preliminary tests

Tables 4.5 and 4.6 show the outcome with small instances, and increasing number of zones, respectively with equity as risk sharing and equity as Rawls's principle. Rows have to be considered by couples: one row shows the outcome considering equity constraints and the following one without equity constraints. Tables 4.7 and 4.8 show the result with increasing cardinality till to medium size instances.

Test	Card.	Gr.d.	Cap.	Comm.	Zones	Equity	cpu time	Total damage
1	10	0.5	10	2	3	yes	0.010998	1456.08
	10	0.5	10	2	3	no	0.012998	894.356
2	10	0.5	10	2	4	yes	0.018997	427.923
	10	0.5	10	2	4	no	0.019996	375.871
3	10	0.5	10	2	5	yes	0.012998	2101.45
	10	0.5	10	2	5	no	0.013997	894.427

Table 4.5: CPU time and total damage for equity as risk sharing, with an increasing number of zones

Test	Card.	Gr.d.	Cap.	Comm.	Zones	Equity	cpu time	Total damage
1	10	0.5	10	2	3	yes	0.001999	8975.24
	10	0.5	10	2	3	no	0.002999	894.356
2	10	0.5	10	2	4	yes	0.001999	375.871
	10	0.5	10	2	4	no	0.001999	375.871
3	10	0.5	10	2	5	yes	0.004999	1562.6
	10	0.5	10	2	5	no	0.004999	894.427

Table 4.6: CPU time and total damage for equity as Rawls' principle, with an increasing number of zones

Test	Card.	Gr.d.	Cap.	Comm.	Zones	Equity	cpu time	Total damage
1	10	0.5	10	3	4	yes	0.025996	866.045
	10	0.5	10	3	4	no	0.027995	677.997
2	20	0.5	10	3	4	yes	0.314952	383.153
	20	0.5	10	3	4	no	0.317951	313.265
3	30	0.5	10	3	4	yes	3.34749	400.658
	30	0.5	10	3	4	no	3.35249	245.669
4	40	0.5	10	3	4	yes	16.6485	289.919
	40	0.5	10	3	4	no	16.6565	246.897
5	50	0.5	10	3	4	yes	57.9702	1098.87
	50	0.5	10	3	4	no	57.9992	1062.44

Table 4.7: CPU time and total damage for equity as risk sharing, with increasing cardinality (threshold for sum of risk difference between zones: 10), see Fig. 4.2

Test	Card.	Gr.d.	Cap.	Comm.	Zones	Equity	cpu time	Total damage
1	10	0.5	10	3	4	yes	0.004999	728.432
	10	0.5	10	3	4	no	0.005999	677.997
2	20	0.5	10	3	4	yes	0.004999	313.265
	20	0.5	10	3	4	no	0.005999	313.265
3	30	0.5	10	3	4	yes	0.004999	327.509
	30	0.5	10	3	4	no	0.008998	245.669
4	40	0.5	10	3	4	yes	0.011998	250.84
	40	0.5	10	3	4	no	0.016997	246.897
5	50	0.5	10	3	4	yes	0.033994	1381.43
	50	0.5	10	3	4	no	0.045993	1062.44

Table 4.8: CPU time and total damage for equity as Rawls' principle, with an increasing cardinality (threshold for risk of most disadvantaged zone: 30), see Fig. 4.3

The results we got, show that both kinds of equity have a negative impact on the damage and make it grow, that is the awaited outcome.

#### 4.5.2 Comparison

The aim of these set of tests is to establish which idea of equity makes total damage increase most. Nevertheless, we can not compare them directly since parameters  $R_D$  and  $R_P$  have not the same meaning. In order to answer this question, we have

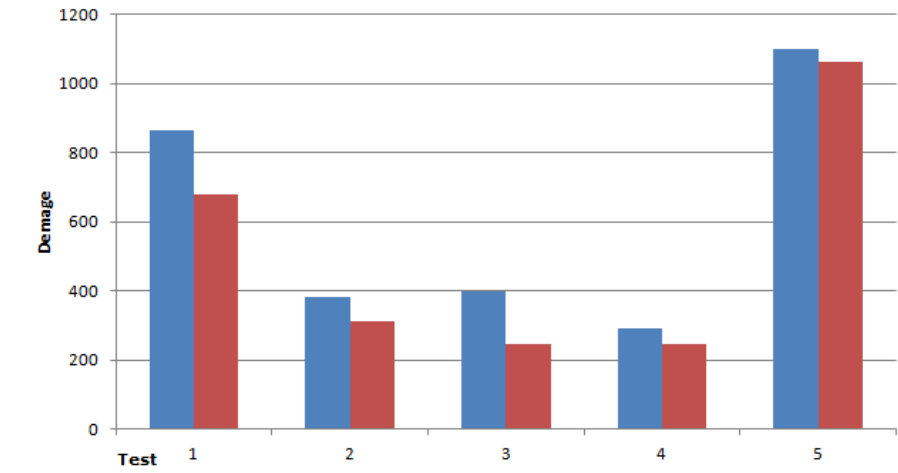


Figure 4.2: Total damage for equity as risk sharing, with increasing cardinality, for details see Table 4.7

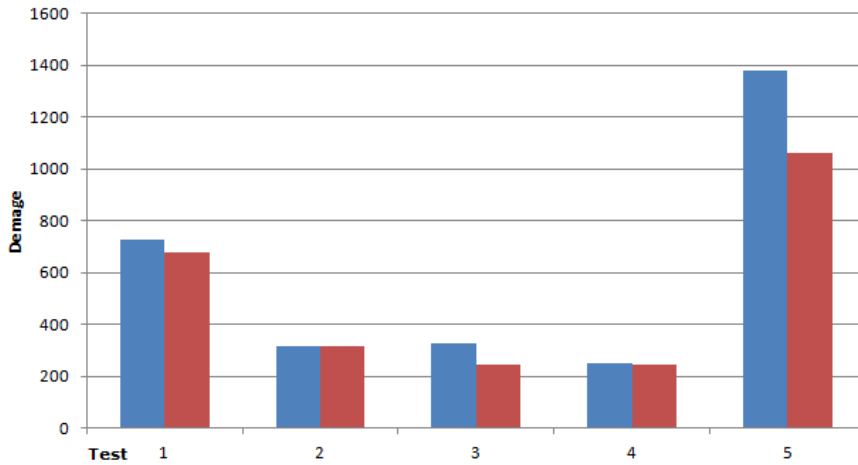


Figure 4.3: Total damage for equity as Rawls' principle, with an increasing cardinality, for details see Table 4.8

to normalize the comparison to equal levels of equity and to map the increase of total damage to the *share* of equity instead of its absolute value. We observe what happens when equity varies from zero to “top”, i.e. to maximum possible level. At zero-equity level, we solve the problem with no care about equity. In others words we drop the equity constraints. At the top-equity level, we require to be as equitable as possible, namely we set the bounds on constraints to tightest possible level, before transportation become impossible (i.e. when a smaller threshold for sum of risk difference between zones or a smaller threshold for risk of most disadvantaged zone would lead to infeasibility of our MP formulation). We scale the range between this two extremes from 0% to 100%. We measure the increment of damage while equity varies from its possible minimum to its possible maximum and we map it on the share of equity. For example, we compare the cost we get when we have the peak

of equity in Risk Sharing and Rawls sense, then when we get the 90% of equity, the 80% . . . and so on.

Tables 4.9 and 4.11 show the value of total damage in function of equity rate, respectively for equity as risk sharing and equity as Rawls's principle. Tables 4.10 and 4.12 focus on the difference between transportation with and without equity constraints, in function of equity rate, respectively for equity as risk sharing and equity as Rawls's principle. Figures 4.4 and 4.5 show graphically these results.

Card.	Gr.d.	Cap.	Comm.	Zones	Equity	Equity rate	Total damage
50	0.5	10	3	4	yes	100; 0%	1062.44
50	0.5	10	3	4	no	100; 0%	1062.44
50	0.5	10	3	4	yes	90; 10%	1062.92
50	0.5	10	3	4	no	90; 10%	1062.44
50	0.5	10	3	4	yes	80; 20%	1064.86
50	0.5	10	3	4	no	80; 20%	1062.44
50	0.5	10	3	4	yes	70; 30%	1066.99
50	0.5	10	3	4	no	70; 30%	1062.44
50	0.5	10	3	4	yes	60; 40%	1069.52
50	0.5	10	3	4	no	60; 40%	1062.44
50	0.5	10	3	4	yes	50; 50%	1072.36
50	0.5	10	3	4	no	50; 50%	1062.44
50	0.5	10	3	4	yes	40; 60%	1075.77
50	0.5	10	3	4	no	40; 60%	1062.44
50	0.5	10	3	4	yes	30; 70%	1082.47
50	0.5	10	3	4	no	30; 70%	1062.44
50	0.5	10	3	4	yes	20; 80%	1090.37
50	0.5	10	3	4	no	20; 80%	1062.44
50	0.5	10	3	4	yes	10; 90%	1098.87
50	0.5	10	3	4	no	10; 90%	1062.44
50	0.5	10	3	4	yes	0; 100%	1108.38
50	0.5	10	3	4	no	0; 100%	1062.44

Table 4.9: Increase of damage induced by equity as risk sharing (normalized)

## 4.6 A heuristic for large-scale instances

As instance graphs sizes and densities grow, the solution of the MP formulation gets harder. Thus, we try to reduce the size of the graphs preserving their fundamental features, in order to look for approximated solutions. The basic idea is to establish a ranking among vertices. Once vertices are classified, a smaller graph can be obtained by removing the least relevant vertices (this is an implementation, in this specific context, of the “abstraction” technique mentioned in Section 1.3.1).

Equity rate	Total damage	Increase
0%	1062.44	0.00
10%	1062.92	0.05
20%	1064.86	0.23
30%	1066.99	0.43
40%	1069.52	0.67
50%	1072.36	0.93
60%	1075.77	1.25
70%	1082.47	1.89
80%	1090.37	2.63
90%	1098.87	3.43
100%	1108.38	4.43

Table 4.10: Difference of damage in function of the rate of equity, in the case of equity as risk sharing

Card.	Gr.d.	Cap.	Comm.	Zones	Equity	Equity rate	Total damage
50	0.5	10	3	4	yes	76; 0%	1062.44
50	0.5	10	3	4	no	76; 0%	1062.44
50	0.5	10	3	4	yes	69; 10%	1068.21
50	0.5	10	3	4	no	69; 10%	1062.44
50	0.5	10	3	4	yes	62; 20%	1076.53
50	0.5	10	3	4	no	62; 20%	1062.44
50	0.5	10	3	4	yes	55; 30%	1088.03
50	0.5	10	3	4	no	55; 30%	1062.44
50	0.5	10	3	4	yes	47; 40%	1105.22
50	0.5	10	3	4	no	47; 40%	1062.44
50	0.5	10	3	4	yes	40; 50%	1124.67
50	0.5	10	3	4	no	40; 50%	1062.44
50	0.5	10	3	4	yes	33; 60%	1210.34
50	0.5	10	3	4	no	33; 60%	1062.44
50	0.5	10	3	4	yes	26; 70%	14197.9
50	0.5	10	3	4	no	26; 70%	1062.44
50	0.5	10	3	4	yes	19; 80%	47472.8
50	0.5	10	3	4	no	19; 80%	1062.44
50	0.5	10	3	4	yes	11; 90%	230180
50	0.5	10	3	4	no	11; 90%	1062.44
50	0.5	10	3	4	yes	3; 100%	381861
50	0.5	10	3	4	no	3; 100%	1062.44

Table 4.11: Increase of damage induced by equity as Rawls' principle (normalized)

#### 4.6.1 Graph reduction through centrality erosion

There are several metrics that can be used to calculate the importance of a vertex in a graph. In particular, sociology has studied this matter deeply, due to its interest for social networks, and has proposed many measures. The metric we use in this section is *centrality betweenness* that considers how often a vertex is along the shortest path

Equity rate	Total damage	Increase
0%	1062.44	0.00
10%	1068.21	0.54
20%	1076.53	1.33
30%	1088.03	2.41
40%	1105.22	4.03
50%	1124.67	5.86
60%	1210.34	13.92
70%	14197.9	> 1000.00
80%	47472.8	> 1000.00
90%	230180	> 1000.00
100%	381861 0	> 1000.00

Table 4.12: Difference of damage in function of the rate of equity, in the case of equity as Rawls principle

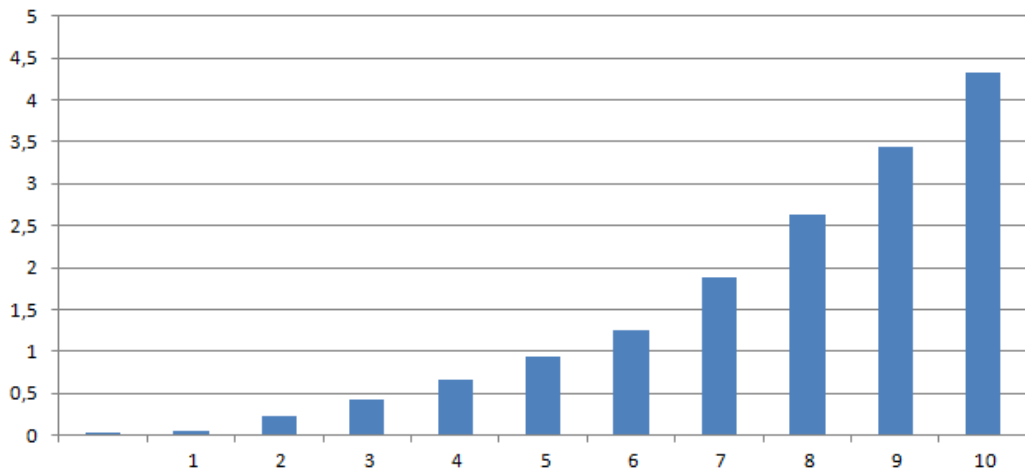


Figure 4.4: Difference of damage in function of the rate of equity, in the case of equity as risk sharing.

between two other vertices. Anthonisse’s work [7] and Freeman’s works [64, 65] are seminal. We refer to them as “traditional” approach. Nevertheless, we consider a more recent work from Brandes [21] (on which this section is based). We introduce below, more formally, *centrality betweenness* and some other common measures.

A *walk* is an alternating sequence of vertices and edges. A walk is closed if its first and last vertices are the same, otherwise it is open. A *path* is an open walk. When no other attribute is specified, it is assumed that it is simple, i.e. that no vertices and no edges are repeated. The *length* of a path is the sum of the weights of its edges. The *distance* between vertices  $s$  and  $t$  is the minimum length of any path connecting  $s$  and  $t$  in  $G$  and we denote it with  $d_G(s, t)$ . We call  $\sigma_{st}$  the number of the shortest paths from vertex  $s$  to vertex  $t$  and  $\sigma_{st}(v)$  the number of shortest paths from vertex  $s$  to vertex  $t$  that pass through  $v$ . We call  $\delta_{st}(v) = \frac{\sigma_{st}(v)}{\sigma_{st}}$  the *pair-dependency* of a pair  $(s, t) \in V$  on an intermediary  $v \in V$ .

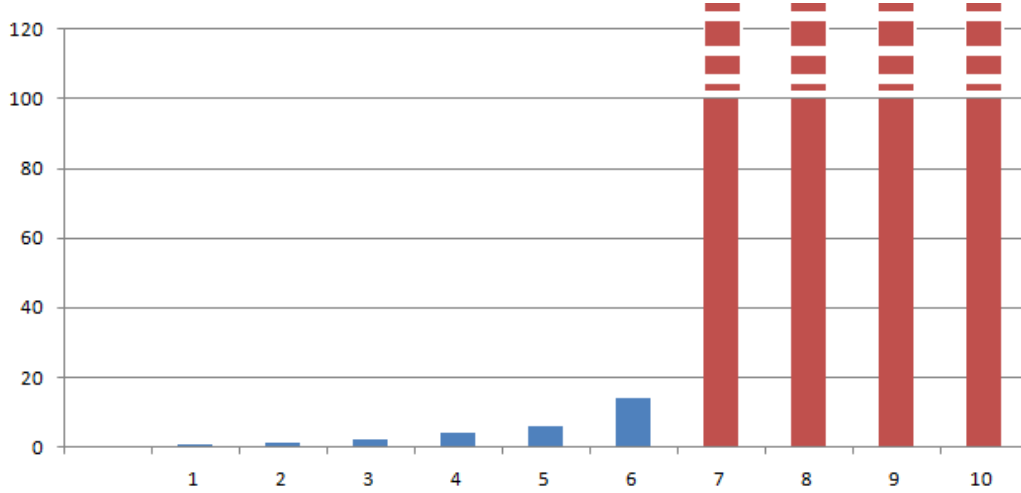


Figure 4.5: Difference of damage in function of the rate of equity, in the case of equity as Rawls' principle. Dotted red bars mean out-of-scale values.

- Closeness centrality (Sabidussi, 1966)

$$C_C(v) = \frac{1}{\sum_{t \in V} d_G(v, t)} \quad (4.20)$$

- Graph centrality (Hage and Harary, 1995)

$$C_G(v) = \frac{1}{\max_{t \in V} d_G(v, t)} \quad (4.21)$$

- Stress centrality (Shimbel, 1953)

$$C_S(v) = \sum_{s \neq v \neq t \in V} \sigma_{st}(v) \quad (4.22)$$

- Betweenness centrality (Freeman, 1977; Anthonisse, 1971)

$$C_B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} \quad (4.23)$$

In order to calculate centrality, we have to consider each combination of vertices in the graph and to find the shortest path between them. Each vertex that is along a shortest path gets a point. The resulting scores for each vertex in the ratio between the total number of shortest paths from node to node and the number of paths that pass through, and represents its *betweenness centrality*.

Typically, *betweenness centrality* is determined through a two steps procedure. First, we compute length and number of shortest paths between all pairs of ver-

tices, then sum all pair-dependencies. From a computational time complexity point of view, traditional algorithms, which are based on the Floyd-Warshall algorithm, belong to  $\Theta(n^3)$ . Brandes introduces a faster algorithm for *betweenness centrality* computation that belongs to  $O(nm + n^2 \log n)$  [21, 22]. We implemented it<sup>1</sup> in order to produce a ranking of the vertices of a given instance and applied it to an instance that we can not solve directly with the method exposed in Section 4.5, due to its size. It is the instance that corresponds to the parameters vector  $(70, 0.5, 10, 3, 4)$ , where, in particular, 70 is the cardinality of the set of vertices. Thus, we calculate the *centrality betweenness* of its vertices. Figure 4.6 provides a graphical representation of the instance graph and of the centrality of the vertices (which are depicted in the figure with different color and size, depending on their centrality). Consequently,

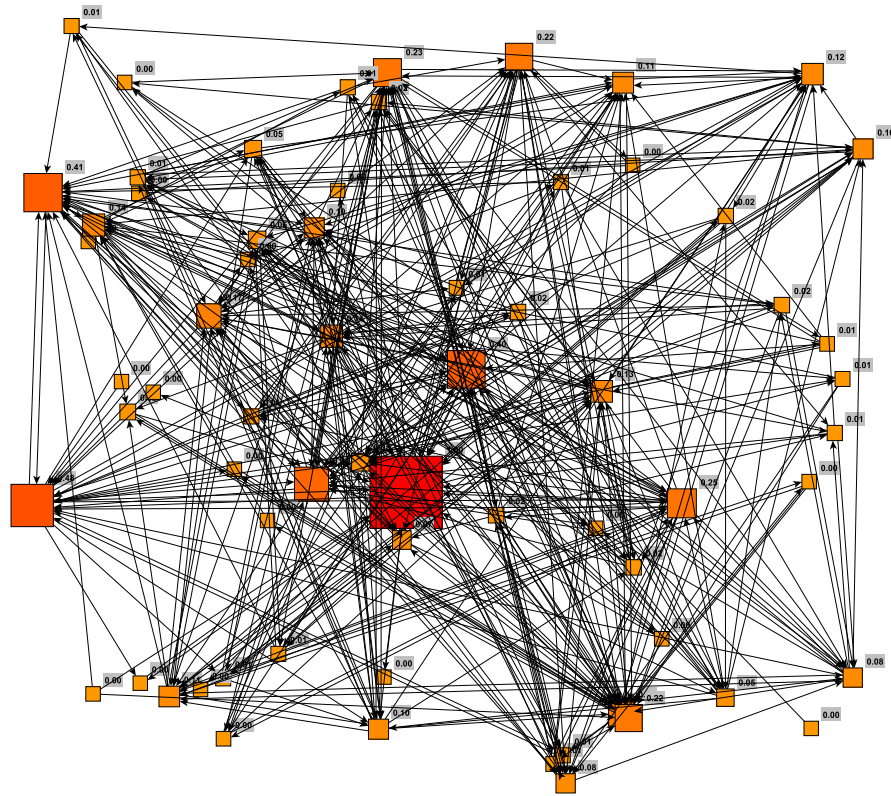


Figure 4.6: Graphical representation of the instance  $(70, 0.5, 10, 3, 4)$  showing the most central nodes. Big red squares have an high betweenness centrality.

we transform the graph corresponding to the instance  $(70, 0.5, 10, 3, 4)$  into a new reduced one, removing the ten less central vertices and considering the graph induced by the remaining vertices. We then solve the problem for this reduced instance, as explained above. Table 4.13 and Table 4.14 show the outcome we obtain with the reduced instance respectively with equity as risk sharing and equity as Rawls's

<sup>1</sup>We exploit the public library GraphStream that is hosted by the University of Le Havre and has been initiated and maintained by members of the  $RI_2C$  research team from the LITIS computer science lab, <http://graphstream-project.org/>.



Test	Card.	Gr.d.	Cap.	Comm.	Zones	Equity	cpu time	Total damage
original	70	0.5	10	3	4	yes	exit code	exit code
	70	0.5	10	3	4	no	exit code	exit code
reduced	70	0.5	10	3	4	yes	309.803	146.152
	70	0.5	10	3	4	no	309.847	145.569

Table 4.13: CPU time and Total damage for equity as risk sharing, for reduced instance (threshold for sum of risk difference between zones: 10)

Test	Card.	Gr.d.	Cap.	Comm.	Zones	Equity	cpu time	Total damage
original	70	0.5	10	3	4	yes	0.051992	145.569
	70	0.5	10	3	4	no	0.081987	145.569
reduced	70	0.5	10	3	4	yes	0.044993	145.569
	70	0.5	10	3	4	no	0.072988	145.569

Table 4.14: CPU time and Total damage for equity as Rawls' principle, with an increasing cardinality (threshold for risk of most disadvantaged zone: 30).

principle. The procedure enables the handling of an instance that we could not solve directly by means of the solver, in the case of equity as risk sharing, and speed up the process in both cases.

Notice that, no matter how we define equity, we are considering equity between regions. Moreover, a same area may have different partitions that are nested (this could make the problem of equitable transportation even more complicated than the scenario we have considered). Typically equity is required by public institutions that operate at different levels and that could be not totally consistent. For example, we can imagine a problem of transport in Europe which shall respect a constraint of equity between nations, but also between regions of a same nation, districts of a same region, cities of a same district and so on. There are different approaches for such a situation. If we consider the subdivision as given, we have to solve an optimization problem, respecting the input data. Otherwise, if we consider the problem from the point of view of the lawmaker, namely if we are in charge of choosing the structure of the administrative zones (typically, within certain limitations) then we face a design problem: we look for the architecture of the transport system that entails the minimum risk and maximum equity. In other words we design the instances. The methodology that we have presented can be useful for both kinds of approach, to search optimal solution for given scenarios and to test different zone subdivisions when we are free to manipulate input data.

## Part III

# Meta-theory



## Some epistemological and historical remarks

In this chapter we provide epistemological and historical remarks that concern some of the subjects considered in this work. This is consistent with the subjective opinion of the author that science and co-science (i.e. philosophy of science) can cooperate fruitfully. However, we think that it is confusing to mix them vaguely, thus we have separated the treatment of the problems presented so far and the consideration about their philosophical implications. Using a terminology that philosophers love, the previous parts represent the object level of this work, this part represents the meta-level.

### 5.1 Philosophy of Engineering

The match between philosophy and engineering is quite unusual and merits further clarifications. The basic issue in the philosophy of science can be introduced as follows: scientists study the world, philosophers of science study how they do that (and sometimes they also study scientists themselves).

Borrowing from Lipton [110],

*“I am a philosopher of science: what do I do? Here is the short version: astronomers study the galaxies; I study the Astronomers.”*

There has been a certain disregard by philosophers of science towards technology, which they consider a straightforward application of pure sciences.

*“The method and the theories of science can be applied either to increasing our knowledge of the external and the internal reality or to enhancing our welfare and power. If the goal is purely cognitive, pure science is obtained; if primarily practical, applied science. Thus, whereas cytology is a branch of pure science, cancer research is one of applied research.” [31]*

This idea, i.e. “technology is applied science”, hides the fact that technology and engineering disciplines have some features needing a special epistemological investigation. Recently, a new branch of epistemology called *philosophy of engineering* has attracted increasing interest: it is concerned with the clarification of the epistemological role of technology and engineering within science and human knowledge. To continue the analogy introduced above, engineers study how to design systems, philosophers of engineering study how they do that. The crucial difference between engineers and scientists is that the former *decide* how to manufacture or produce working artifacts and systems, while the latter *analyze* nature and formulate theories to explain how natural systems work. Decision-making naturally brings engineers to think about objectives much more than natural scientists need to do. This profiles a different kind of rationality. On the one hand, we might believe that models are simplified versions of reality that exists independently from our ideas about it, and that the task of science is to describe this reality. On the other hand we might think that models do not describe reality, but they actually create it (indeed most of the modern epistemology tells us that all observation is theory-laden, for example see Hanson [81] on this point and, more philosophically, think of Kant and his *Copernican revolution*). Likewise, we might feel a need to simply explain systems, as opposed to endowing them with aims. If truth is not discovered, but it is invented, the hierarchy between science and technology is inverted.

*“Despite the more than two millennia that separate Aristotle’s thinking from ours, Aristotle’s conception [sets] the agenda for almost all subsequent thinking about explanation. [...] The rivalry had been between those who thought that all causal explanation must proceed in terms of efficient causation and those who (following closely on Aristotle’s footsteps) thought that there is room (and need for) teleological explanation (that is, for explanation that cites final causes). [...] Aristotle saw goals and purposes in nature, mechanical philosophers either excised all purpose from nature (Hobbes, Hume) or placed it firmly in the hands of God (Descartes)”. [137]*

This debate fails to have a clear outcome within epistemology, but if the target system is artificial rather than natural, then it must have a goal, and the issue becomes clearer. What we might call the “pure problem” of scientists is “is it true?”, while that of engineers might either be “does it work?”, or, perhaps more appropriately, “does it do what the stakeholders want?” It is clear that there is a relationship between being able to verify a statement and making a choice. However, decision making and systems design have some features that make them a special case from an epistemological point of view.

*“In engineering the ultimate purpose of modeling is to realize reliable artifacts or technical processes. This contrasts substantially with the natural*

*sciences where, conceptually at least, the aim underlying the modeling activities is to gain knowledge for knowledge's sake.” [6]*

Epistemologists, in the first half of the '900, usually made reference to natural sciences as chemistry, biology and, most of all, physics (probably due to its resounding success). In this case, the observer is in front of a system that is given and he/she has to describe and understand it. However, in engineering the system is actually *built* by the observer (or one of his/her fellow humans). Thus, the *demarkation criterion* of natural science may be not perfectly suitable. Systems designers still have to do verifications and observations (as natural scientists) but most of all they have to make choices. They are interested in the truth of statements as much as in the effectiveness of choices. From the point of view of systems design, good models are the ones that help to split properly the domain of possible choices in good and bad ones. Some epistemologists underline the problem-solving aspect of science, for example Laudan.

*“Science is essentially a problem-solving activity. [...] The approach taken here is not meant to imply that science is “nothing but” a problem-solving activity. Science has a wide variety of aims [...] My approach, however, contends that a view of science as a problem-solving system holds out more hope of capturing what is most characteristic about science than any alternative framework has.” [100]*

Considering science as problem-solving corresponds to a change of perspective since we are more interested in getting local solutions rather than global theories. In particular, Khun suggested Operations Research as a good example of the problem-solving approach to science. This is remarkable since in our work we have used extensively the methods provided by OR, tackling the problems presented in each chapter through Mathematical Programming.

*“For Kuhn, science is problem-solving rather than truth-seeking activity . . . . And what would be a more striking example of problem-solving than OR! . . . As a problem-solving activity OR is oriented towards practice: it tries to use the methods of science to find optimal solutions to problems concerned with alternative courses of actions. As the solutions are its primary aim, it is clear in which sense OR is not a truth-seeking activity: it is not a knowledge-seeking enterprise.” [129]*

Philosophy of engineering focus of these special aspects of applied science. We adopt the same perspective. For each engineering application discussed in the chapters of this thesis, we provide some elements of the philosophical debate which helped to settle the current definition of the problem, and propose some epistemological considerations. We remark that the structure of this work splits clearly the section

dedicated to science and this section dedicated to methodological and epistemological considerations.

It has been said that “*Philosophy of science is about as useful to scientists as ornithology is to birds*”<sup>1</sup>, namely that it is not very useful in practice, but we try to show that some epistemological issues arise anyhow. In our opinion, they require a consideration. At least, epistemology is useful for an external analysis of the scientific method. A scientific analysis of the scientific method would be self-referential.

Nevertheless, no-one is better placed than an scientist or an engineer to understand and analyze his or her own way of working. This is why this thesis is, above all, a scientific work. With the words of Schlick,

*“A philosopher, therefore, who knew nothing except philosophy would be a knife without blade and handle. Nowadays a professor of philosophy very often is a man who is not able to make anything clearer, that means he does not really philosophize at all, he just talks about philosophy or writes a book about it. This will be impossible in the future. The result of philosophizing will be that no more book will be written about philosophy, but all books will be written in a philosophical manner.”* [155]

### 5.1.1 Epistemic vs non-epistemic values

McMullin [117] introduces a distinction between *epistemic* and *non-epistemic* values, that is relevant in epistemology. He proposes that a value is epistemic if it helps to “*promote the truth-like character of science*”. Otherwise, it is non-epistemic. Dorato [50] confirms that we can use the term *epistemic* for values “*regarded as capable of furthering our knowledge*” and *non-epistemic* to refer essentially to values that are ideological, economical, political, ethical, environmental, esthetic or religious. Non-epistemic values can influence science, indirectly. They influence, for example, the choice of the destination of economic endorsement of research projects. Nevertheless, there is a strong agreement, in the scientific community, on the idea that non-epistemic values have no role in determining scientific truth. Non-epistemic values influence the use of the results of pure science, but are never (or hardly never) integrated in the content of scientific theories.

However, for engineering and technology disciplines the role of non-epistemic values appears to be less clear. Safety, equity and economical sustainability are examples of non-epistemic values (since they do not produce knowledge) that have an important role in the decision making process concerning real systems. Engineers, who have to choose between two or more alternative models, in some cases, have to consider non-epistemic values, and integrate them in their models. It is the case of the systems we have considered in this work.

---

<sup>1</sup>Richard P. Feynman

This leads us to think about the way a model can gain a justification when it does not rely upon pure epistemic values. In fact, in Chapter 3 we have adopted an experimental approach. We “try and observe”. This is not unusual, but in Chapter 2, we have adopted a collaborative methodology, taking care of business needs. Moreover, in Chapter 4 we have integrated equity in our models. This is not totally common. Thus, in the next sections, we present the tradition that is behind each one of these approaches.

However, preliminarily, we present some remarks about the concept of *model*, since we have tackled several problems with a model oriented approach using mathematical programming (see Section 1.2) as unified language to formalize them. Thus, independently from specific applications, modeling *tout court* has got a subject of our reflection, and *model* is one of the keyword of this work. Moreover, from an epistemological point of view, some questions arise naturally.

- What we can properly consider as a “good” model of a given system from a scientific point view?
- Given a model  $M_a$  and model  $M_b$  both referring to a same target system  $S$ , when we can state that  $M_b$  is better than  $M_a$  ?
- What does it mean for a model to be better than another model ?
- What does it mean for two or more models to refer to a same target system ?

## 5.2 Models

The root of the term *model* can be traced back to the Latin term *modus* which in turn would derive from the Indo-European root “med-”. Its meaning is measure [159]. *Modus* has two diminutives *modellus* and *modulus* which we find in different contexts linked to engineering related disciplines. The roman architect Vitruvius uses *modulus* to mean architectural standard, which is a surprisingly modern use of the term. Tertullianus uses *modulus* to indicate basis for a marble sculpture. In the period which spans from the Roman Empire to the Middle Ages, terms derived from *modulus* spread across Europe and we detect the terms *modle*, *mole* and *moule*, which came into English as *mould*. Modern English also introduced directly the term *module* from Latin. During the italian renaissance *modelo* and *modello* are employed by important architects, such as Brunelleschi, who uses it while building the cupola of the dome of Firenze, and Alberti:

*“Be sure to have a complete Model of the Whole, by which examine every minute Part of your future Structure eight, nine, ten Times over, and again, after different Intermissions of Times”. [5]*



From the Italian *modello* derives the French *modèle* and the English *model* and *modell*. Shakespeare uses *model* both with reference to buildings, thus in the architectural sense, and in a more general sense as “kind of behavior” and Bacon indicates with *modulus* a mental copy of the real world, which is quite close to the modern use. Nowadays these terms are intensively diffused. For example, during the decade 1990-1999 there have been 17,000 publications including them in the title.

The remarkable point is that along centuries there is an interesting feature which characterizes models: they appear to be tools which help to design artifacts. Models are visions of a target system constructed respecting constraints drawn from its environment, which help the system designer/architect to conceive it. In engineering disciplines, modeling is first of all an activity that is close to design. The designing of systems and services requires both analytical and synthetic processes, because designers invent and create new *artificial* systems to fulfill a need. This is different from describing and understanding a given *natural* system. From this point of view modeling assumes a meaning which is much more practical with reference to other scientific disciplines as natural sciences, formal logic and mathematics. Modeling is a set of activities, tools, heuristics (in the broad sense of the term), capabilities which lead a designer to build system-answer to a problem-question which he/she is confronted to.

### 5.2.1 Model validation

*“The mathematical models that are used in OR are representations of the system under study. These models may be imperfect and idealized, but still the quality of the solutions that they yield crucially depends upon their closeness to reality in the relevant respects.” [129]*

Engineers separate the “judgment” of a system into two distinct phases, *verification* and *validation*. The verification process guarantees that the system has been realized correctly, respecting all the specifications documented during the phase of requirements engineering. The validation process ensures that the system functions as expected. Notice that, from an end-user perspective, a system which performs perfectly a wrong task is not a good outcome. This issue is very important in systems design.

*“Simply put, the Product Verification Process answers the critical question - Was the end product realized right? The Product Validation Process addresses the equally critical question - Was the right end product realized?” [88]*

This issue concerns also the method of OR, which typically includes two phases: in the first phase a problem is formalized into a model; in the second phase efficient techniques are searched in order to solve the model. Model verification deals with

questions about the capacity of providing correct solutions with a limited amount of computational resources and time. We refer to this issue as the problem of *efficiency*. Model validation assesses that the model really addresses the right problem. We refer to this issue as the problem of *effectiveness*. For example, a model for the shortest path problem and a fast and correct algorithm that finds its solutions would not be a good answer for someone who is looking for paths that go through the “top  $n$ ” interesting cities starting from Milan and arriving in Paris. It would be efficient but not effective. In OR several important problems are already accurately identified and classified, therefore the focus is most of all on the capacity of solving them, i.e. efficiency. The problem of efficiency is well defined. Computational complexity theory deals with it and provides a stable framework (see Section 1.2.4). However, engineers and system designers are often puzzled by the problem of writing the right model. In systems design effectiveness is a major issue.

*“- What is a valid model? - has been one of the least discussed topics in the OR literature. [...] Thinking about model construction and model validation is basically to raise the issue of different ways of producing knowledge and deciding about the acceptability of the knowledge thus produced”. [99]*

The problem of effectiveness encompasses several approaches and has blurred boundaries. Validation tests can be based on comparing model predictions to real world results. However this kind of validation is not always possible because repeated tests can be expensive, time-consuming or simply impossible. Thus, alternatively, models can be validated using historical events and inter-subjective arguments. In our opinion, the problem of model validation in OR can not be separated from general issues about the approach to scientific knowledge. We believe that philosophy of science and in particular philosophy of engineering are good frameworks for the problem of effectiveness. A few authors share this opinion with us.

*“Whether Operational Researchers are aware of it or not does not make any difference: to take an option in the debate on model validation in OR is, explicitly or not, to actualize epistemological choices”. [52]*

### 5.3 Experimental approach to scientific knowledge

In Chapter 3 we proposed different models of a recommender system and to establish which is the best one we have *tested* them. This appears to be reasonable, even obvious. Modern science is empirical. Experimentation has a role in science which can not be underestimated. According to R.P. Feynman:

*“The principle of science, the definition, almost, is the following: The test of all knowledge is experiment. Experiment is the sole judge of scientific truth” [140]*

Nevertheless, in this section we provide some arguments to remind that the debates that emerged in contemporary epistemology show that the role of experimentation is (sometimes) considered as troublesome. There are a bright and a dark side of the coin. We start from the bright side.

First of all, experiments are used to produce a *confirmation*, as they can give us strong arguments to trust a hypothesis. Secondly they can favor the *discovery* of new theories showing new unknown phenomena which call for an explication. As representatives of these two uses of experiment, we can cite, among others, G. Galileo and F. Bacon. Both of them championed a more empirical attitude in natural philosophy and both of them supported a new vision of knowledge based on observations that had to be performed without prejudice or preconception. However, we consider Galileo to exhibit an example of the use of experimentation to confirm a theory and Bacon as an example of use of experimentation to favour the discovery of new theories.

Observations can endorse a theory. With the telescope, Galileo discovered the four large moons of Jupiter, which, since they do not orbit Earth, provide an argument against the Ptolemaic theory that fixed it at the center of the universe. In this case, facts obtained through experimental work (repeated observation) confirm a theory (Copernican system).

Observations can foster new, general ideas, as explained by Bacon. In fact, Bacon was a convinced inductivist. His *Novum Organum* (1620) can be considered as the first modern work on inductive logic. In particular, it analyses the methods that can be used to produce theoretical inductive inferences, namely from particular to general, which had been relegated to a minor role during the previous centuries.

*“The syllogism consists of propositions, propositions consist of words, and words are tokens for notions. Hence if the notions themselves (this is the basis of the matter) are confused and abstracted from things without care, there is nothing sound in what is built on them. The only hope is true induction.”*

More recently, the more radical defense of empiricism is reasserted by the logical empiricists of Vienna Circle<sup>2</sup>: who stated, in their *Manifesto*, that true knowledge is totally empirical because the scientific enterprise is characterized

*“essentially by two features. First it is empiricist and positivist: there is knowledge only from experience [...] Second, the scientific world-conception is marked by the application of a certain method, namely logical analysis.”*

---

<sup>2</sup>The Vienna Circle was an association of philosophers centered at the University of Vienna in 1922. Among its members there were Moritz Schlick, Rudolf Carnap, Richard von Mises, Otto Neurath, Herbert Feigl.

One of their most famous thesis is the *verification criterion of meaning*: the meaning of a proposition consists in its method of verification, and a proposition which cannot be verified is meaningless. Thus, the role of experimental verification is even stronger than in the vision of Galileo and Bacon, since it is at the basis of meaning.

We now take a look at the dark side of the experimentation coin. Duhem [54] proposes that it is not possible to test experimentally a single hypothesis because complex theories includes many hypotheses and it is really hard to establish which statements are contradicted by a test (systems engineers would call this a traceability problem). Moreover, an observation that refutes a model can be compatible with many other ones. For example, the observation of Galileo was consistent with both the models proposed by Copernicus and the one proposed by Tycho Brahe. This position is known, nowadays, as Duhem-Thesis<sup>3</sup>.

A second difficulty concerns the trustworthiness of what we are used to consider *objective facts*. Starting from the platonic *allegory of the cave* up to now, several philosophers have warned about the possibility that facts could be illusory. Many times in the history of philosophy evidence has been called into question. However, in this case, the target is not knowledge in general, it is the exactly the scientific method which is questioned. In the context of modern science a common reference, from this point of view, is the work of Hanson, as mentioned above. Hanson believes that there is not unconditioned observation of facts and, moreover, there is not a neutral language to express them. Observational terms are “full of theory”. Thus the idea that theories are confronted to pure facts is wrong, in his opinion.

*“There is a sense, then, in which seeing is a ‘theory-laden’ undertaking. Observation of  $x$  is shaped by prior knowledge of  $x$ . Another influence on observations rests in the language or notation used to express what we know, and without which there would be little we could recognize as knowledge.” [81]*

What we observe is influenced, from the beginning, by our system of reference, our opinions, our background knowledge and, in general, our theory.

A third difficulty is explained by Hempel. He proposed the so-called paradox of confirmation, which he explains through the example of the ravens. We normally admit that the observation of a black raven confirms the hypothesis that “all ravens are black”. On the other hand, a white raven is a clear counterexample. However if we also admit (and in general we do) the *equivalence condition*, then we get strange results. The *equivalence condition* states that if two hypothesis are logically equivalent, then certain evidence that confirms the first one confirms also the second (equivalent) one. A logical equivalent of “all ravens are black” is “all non-black objects

---

<sup>3</sup>We remark the often the terms Duhem-Thesis and *Duhem-Quine Thesis* are used as equivalent, but, in reality they refer to quite different thesis.

are non-ravens”. This last is confirmed by a non-black non-raven, e.g. a white tie. It follows that a white tie also confirms “all ravens are black”. This is logically correct, but it sounds strange.

We know that Popper proposes a fundamental improvement to the verification principle of Vienna Circle. He believes that inductive inferences have no justification, since no matter how many singular facts you have observed, you are never sure that a different singular phenomenon could occur, making your general conclusion wrong. Thus verification is, in practice, not feasible. He introduces a different criterion to defend the possibility of empirical justification of a theory. A theory has to divide the world into two distinct classes of phenomena: the ones that are compatible with it and the ones that contradict it. Thus, we should not look for facts that confirm a theory, but for the ones that could make it false. The longest a theory resists to these assaults, the better. It is trusted, or, using his terminology, *corroborated*. This is a considerable progress with reference to the positions of Vienna Circle. Problems caused by induction are reduced.

Nevertheless, according to his opponents, the falsification method proposed by Popper does not escape to the issues of theories underdetermination. During the sixties, authors like Kuhn and Lakatos promoted the idea that science progresses through many different ways, making our comprehension of its method more encompassing. Their focus was no more on one single theory against facts. Scientific research started to be considered as a complex system that comprehends many heterogeneous elements. The terms *paradigm* proposed by Kuhn and *research program* proposed by Lakatos gained a remarkable success and entered the terminology of philosophy of science, becoming quite common. In particular (following [97,98]) there are 4 types of basic research programs: *descriptive*, *explanatory*, *design*, *explicative*. Descriptive research programs aim “simply” to describe of a set of phenomena, while explanatory programs try to provide an explanation and a framework to predict similar phenomena. These first two types concern empirical sciences. Design research programs deal with the realization of artifacts that fulfill certain previously chosen needs. This type concerns engineering and related disciplines. Explicative research programs are meant to provide precise, possibly formal explication of interesting, but unclear concepts. This last type regards mathematics and analytic philosophy. Thus, there are at least four different approaches to science, and not all of them are purely based on experimentation. The “lesson” of these philosophers of science is that we should consider the method of science simply as “what scientists do”, without limitations. Feyerabend, most of all, strongly endorses this point of view.

From our point of view, we notice that, actually, system designers and decision makers (sometime) have to make choices that can not be based on experimental evidence. Therefore, in the following sections, we consider different possible approaches.

## 5.4 Collaborative approach to scientific knowledge

The method we have used in Chapter 2 to validate our model is basically a collaborative agreement of a set of practitioners. In this section we trace historical and conceptual roots of this kind of method, namely the search of truth (only) through an open discussion.

There are approaches to the scientific knowledge that skip most of the issues about the capacity of science of catching the ultimate truth about reality. For example, *instrumentalism*.

*“Instrumentalism can be formulated as the thesis that scientific theories, the theories of the so-called “pure” sciences, are nothing but computational rules (or inference rules)”. [135]*

Ontological<sup>4</sup> problems about the effective existence of an immutable “being”, that has to be described by a conclusive explanation, are totally left out. Instrumentalism does not focus on the distinction between truthfulness and falseness of scientific theories. On the contrary it considers, by choice, “only” their practical utility. Important representatives of this approach are, among others, E. Mach, H. Poincarè, P. Duhem, E. Le Roy. For example, Poincarè proposes that we can consider the axioms of the geometry as simple *conventions*. Similarly, Le Roy thinks that science has a pure instrumental value and that scientific laws are only convenient synthesis of sets of facts. The position of Duhem is more variegated, but not very different.

*“A physical theory is not an explanation. It is a system of mathematical propositions which can be derived from a small number of principles that serve to precisely depict a coherent group of experimental laws in a both simple and complete way”. [53]*

The “second”<sup>5</sup> Wittgenstein (see. [174]) believes that a general formal study of the language is not viable. No theory can provide general rules that are valid in all cases. On the contrary, we can establish only local norms since human language is elaborated in local contexts. He thinks that these norms emerge from behaviors and cultures based on what he calls *language games*, i.e. specific sets of linguistic rules. A perfect language does not exist and in particular there is not a perfect scientific language. Moreover, in his opinion, this reflects the absence of a common underlying structure, namely the absence of a common logic. We should drop the idea that there is one single “Logic” at the basis of human rationality and accept the fact that we act and think according to particular *practices* which are functional to particular aims and can not be generalized.

<sup>4</sup>Ontology is the branch of metaphysics that studies the nature of existence or being as such

<sup>5</sup>We remark that the “second” Wittgenstein is almost different from the “first” one, whose positions are represented most of all by the *Tractatus logico-philosophicus*.

Instrumentalism, conventionalism and the “second” Wittgenstein open the door to the entrance in the field of philosophy of science of elements that, in the first decades of the 20<sup>th</sup> century, had been kept out. Social components are introduced as a fundamental part of scientific knowledge. The separation between external and internal components of scientific enterprise starts weakening, so that context and content begin running into one and knowledge is no more *justified true belief*, but, more weakly, *locally accepted belief*. Physics loses its supremacy as model of all scientific disciplines, and the nineteenth-century idea, renewed by the project of unity of science of Vienna Circle, that all branches of science could be reduced to mathematical explanation, is replaced by a more encompassing approach that admits final causes, interpretations, narrative explications. From the point of view of these authors, the study of nature is similar to the study of social institutions, myths, political groups. In other words, these epistemologists think that knowledge is only a social construction, namely that truth does not exist in itself and it is only agreed consensus (often, of experts). This current of thought suggests that what we consider true is composed by simple beliefs that someone, who has the power, prestige or status to do it, has legitimated.

Bloor and Barnes and other researchers of the University Edinburgh funded in the '60 the *Strong program of sociology of knowledge* (*Strong Program*, for short) endorsing these ideas. This stream of research fits in with the tradition of sociology of science of Merton (cf. [119]) but has stronger objectives. Traditional sociology of science wants to explain the influence of social factors on the process that leads to a discovery, but does not believe that they influence also its content. We could say that it focuses more on scientists than on scientific theories. Basically, the contribution of sociology is considered useful to explain scientific failures. Correct theories do not need sociological explanations. Wrong ones can be object of a sociological analysis. On the contrary the *Strong Program* states that truth is a social product, thus all statements, even correct ones, have a sociological justification. For example, Bloor thinks that the psychologist approach to mathematics proposed by J.S. Mill still had full plausibility. Mill thinks that to understand mathematics is equivalent to understand the psychological processes that are carried out by mathematicians. Frege contrasted this idea, asking for an objective substrate of mathematics. Starting from Frege's objections, Bloor states that this substrate is provided by the inter-subjective layer of psychological processes, namely the social one. Mathematics, from this point of view, becomes essentially a social practice.

We remark that, among others, Popper was absolutely opposed to this approach and he believed that sociology and psychology cannot be used to ground science.

*“...to me the idea of turning for enlightenment concerning the aims of science, and its possible progress, to sociology or to psychology ... is surprising and disappointing. In fact, compared with physics, sociology and*



*psychology are riddled with fashions and uncontrolled dogmas . . . This is why I regard the idea of turning to sociology or psychology as surprising.*" [136]

However, independently from the question of establishing which one of these opposed approaches to knowledge is correct (which is not our task) we can retain that there is an approach to scientific knowledge that tells us that a decision can be legitimately supported by a deal stipulated by all the people in charge of the choice.

Coming back to the point of view of our work, we can observe that collaborative decision making has its own tradition and, thus, indirectly, a kind of legitimation. We do not believe that this is the best method, neither that this is the only method, as strong program sociologists tell us. Nevertheless, in practice, when no other options are available, or empirical evidence is missing, decisions are taken by means of stakeholders' agreement. We concur that this is not inadmissible. In practice, it happens, quite often. In our case, the best model in Chapter 2 has been chosen in this way. In our experience, this is not unusual in projects management and systems design.

## 5.5 Ethical approach to scientific knowledge

In Chapter 4 we have considered the possible integration of equity constraints in a MP formulation. Equity deals with ethics. The separation between epistemic and non-epistemic values, in this case, wavers. This calls for a reflection about the plausibility of such an operation. In this section we provide some remarks about the idea of equity that we have exploited and (a few) general observations. We look in literature for relationships between ethics and science (OR in particular).

Churchman [40] warns about the possible immorality of OR which, in his opinion, could not respect the Kant's moral law "*make only those decisions which treat humanity as an end, never as a means only*" since, in some occasions, OR treats people only as means, in order to achieve an optimum. Nevertheless, we have integrated Rawls's moral principles in our model. Philosophically speaking, Rawls is Kantian. His theory fits with the tradition of the social contract of Locke, Rousseau, and Kant.

*"[My] theory . . . is highly Kantian in nature. Indeed, I must disclaim any originality for the views I put forward. The leading ideas are classical and well known. My intention has been to organize them into a general framework. . . "* [143]

Rawls's main work, *A Theory of Justice*, presents organically a set of concepts that he has developed over years. Most of all the concept of "*justice as fairness*", that is based on the "*liberty principle*" and the "*difference principle*". He states again and refines them, in another, later work, *Justice as Fairness: A Restatement*:



1. “Each person has the same infeasible claim to a fully adequate scheme of equal basic liberties, which scheme is compatible with the same scheme of liberties for all . . . .”
2. “Social and economic inequalities are to satisfy two conditions: first, [a] they are to be attached to offices and positions open to all under conditions of fair equality of opportunity; and second, [b] they are to be to the greatest benefit of the least-advantaged members of society (the difference principle)” [146]

Rawls exploits a mental experiment based on the idea of “*veil of ignorance*”. We should imagine to ignore our actual social status and try to establish a set of rules for the whole society. Probably we would choose equitable rules that prevent ourselves from any disadvantages, whatever condition we really have.

Rawls opposes his approach to the three main alternatives that he identifies in *utilitarianism*, *intuitionism*, and *perfectionism*. Utilitarians think that there are no good actions “*in themselves*” since the morality of a course of action is determined by its outcome. In particular they aim at the maximization of overall welfare, with no care of differences between individuals.

“ [ . . . ] utilitarianism is generally held to be the view that the morally right action is the action that produces the most good. [ . . . ] the theory is a form of consequentialism: the right action is understood entirely in terms of consequences produced. [ . . . ] On the utilitarian view one ought to maximize the overall good, that is, consider the good of others as well as one’s own good” [51]

Moral intuitionism proposes that ethics is based on moral laws that are self-evident and true “*a priori*” and we do not have any method to discriminate among them, out of our intuition and our moral sense:

“ Intuitionist theories, then, have two features: first, they consist of a plurality of first principles [ . . . ] second, they include no explicit method, no priority rules, for weighing these principles against one another: we are simply to strike a balance by intuition, by what seems to us most nearly right.” [143]

Perfectionism uses the concept of human perfection as a guideline.

“ Perfectionist ethics has often been associated with elitist doctrines. . . . the perfection that matters the most is the perfection of those who are capable of achieving the most. This “superman” version of perfectionism, a view famously associated with Nietzsche, . . . .” [171]

The relationships between ethics and OR are recurrent. Wenstøp [173] offers us a comprehensive overview of the last four decades, indicating the work of Boulding [20] as a divide. Boulding proposes OR as an instrument for ethics due to its capability of optimizing consequences of a decision and maximizing utility, which is the goal of some kinds of moral approaches, for example utilitarianism.

Ackoff observes that OR should take care of the interest of the stakeholders (an idea that is consistent with the approach we have adopted in this work).

*“Decisions should be made by consensus of all who are directly affected by the decisions, the stakeholders.” [2]*

Wallace’s edited book, *Ethics in Modeling* [172], covers several arguments related to the role of ethics in design disciplines and endorses an attentive care for stakeholders and ethical issues. Brans [24, 25] indicates Multi Criteria Decision Analysis as the OR tool that can *“take the interests of the stakeholders and nature into account, and calls for a multifaceted concept of ethics, consisting of respect, multi criteria management and happiness”* [173]. Gallo [67] underlines that the research should care about both the consequences of a decision and the respect of fundamental principles. He identifies the two that should ground OR. The *responsability* principle, based on the thought of Jonas [87], and the *sharing and cooperation* principle. Brans and Gallo [26] provide another historical account of the relationships between OR and ethics, indicating Churchman as one of the main initiators of this “match”. They observe that:

*“Unlike natural sciences, OR/MS<sup>6</sup> [...] has as its object not natural reality but rather a man-made reality, the reality of man-machine complex systems [...] Hardly any area in OR/MS can be considered far enough from the real world to escape from ethical considerations”.*

Mingers [121] analyses the relationships between OR and *Discourse ethics* (DE), a moral framework developed by Habermas [79, 78]. According to Mingers, this theory fits well with the science of decision-making. Habermas thinks that we can, through the analysis of communicative structures, identify the conditions for the acceptability of a valid argument and that these conditions are common to a valid moral theory.

*“How then should we apply DE to OR? [...] DE does not put itself forward as a panacea but it does provide a processual template against which proposals and decisions can be tested for ethical legitimacy, and, if followed, should lead to actions that are better in the long run for both organizations and civil society as a whole.” [121]*

Le Menestrel and Van Wassenhove focus on the trade-off between

---

<sup>6</sup>Operations Research / Management Science (OR/MS)

*“scientific legitimacy of OR models (ethics outside OR models) and the integration of ethics within models (ethics within OR models)” [101].*

This argument recalls the opposition of epistemic and non-epistemic values introduced previously. They identify three possible attitudes towards the relationships between OR and ethics. The first one corresponds to a sharp separation between them. It ensures objectivity of OR, but, in their opinion, is incomplete. The second one integrates ethics in OR. This approach is more complete, but has the flaw of accepting a certain amount of subjectivity. The third approach is based on a distinction between OR model and OR process. Ethics should be integrated with OR process, and not in the models. The OR process can operate as a connector between OR models and the real world and can include ethical matters without compromising the objectivity of OR models. Thus, they refer to this approach as *ethics beyond the model*.

*“We present three methodological approaches to combine ethics with Operational Research. The first one is ethics outside OR models [...] The second approach is ethics within OR models [...] The third approach is ethics beyond OR models”*

## 5.6 Computational approach to scientific knowledge

Many scientists use software systems in order to build other kinds of systems that are physical. Computer scientists have a different point of view, since software systems represent their target systems. Moreover, a model of a software system is, almost always, another software system. In this case, efficiency appears to be one of the most important criteria to be considered to choose between different models. The models that entail the most efficient computation are good. This approach is, a fortiori, widespread in mathematical programming community. Computational complexity is a well developed field of computer science. In Section 1.2.4 we have provided some elements, but we do not provide a full exposition of this matter. This is out of the scope of this work. We underline only a few points that are philosophically relevant, in our opinion.

In fact, the idea of efficient computation is based on the idea of effective computation (this is not particularly surprising). We think that this second concept is worth an epistemological analysis.

During the 1930s several computational models have been proposed: the universal Turing machine proposed by Turing, the  $\lambda$ -calculus proposed by Church and the  $\mu$ -recursive functions. It was proved that these three models are equivalent. This led to the formulation of a famous thesis (now known as the Church-Turing Thesis).

**Thesis 5.6.1 (Church-Turing Thesis, CT-T)** *Whenever there is an effective method (algorithm) for obtaining the values of a mathematical function, the function can be computed by a Turing Machine.*

The statement of 5.6.1 as a *thesis* has not been totally straightforward. According to [161] it went through Church's definition, Turing's definition, Church's thesis, Turing's thesis and finally Church-Turing thesis<sup>7</sup>. As suggested in [164] the modern terminology is probably originally due to S. Kleene who called *Thesis I* the Church's definition in [90] and used *Church's Thesis* and *Turing's Thesis* in [91]. Finally, if we follow [42], is again Kleene who introduced the expression *Church-Turing thesis*.

*“So Turing's and Church's thesis are equivalent. We shall usually refer to them both as Church's thesis, or in connection with that one of its ... version which deal with - Turing machines - as the Church-Turing thesis.”* [92]

A first unusual implication of this thesis is that it has generated a series of different interpretations. Several authors consider it as a claim that a Turing machine can calculate everything that a computer (of every kind) can calculate. R. Gandy introduced a distinction between the Church-Turing thesis and this second claim, who he called *Thesis M* (followed by other authors) [68].

**Thesis 5.6.2 (Thesis M)** *Whatever can be calculated by a machine (working on finite data in accordance with a finite program of instructions) is Turing-machine computable.*

Opinions about this distinction are not unanimous. For example, we can consider the exchange between J. Copeland and A. Hodges. On one hand, Copeland endorses the distinction.

*“A myth seems to have arisen concerning Turing's paper of 1936, namely that he there gave a treatment of the limits of mechanism and established a fundamental result to the effect that the universal Turing machine can simulate the behaviour of any machine.”* [42]

On the other hand, Hodges thinks that Church actually refers to machines:

---

<sup>7</sup>Church's definition:

“We now define the notion, already discussed, of an effectively calculable function of positive integers by identifying it with the notion of a recursive function of positive integers (or of a  $\lambda$ -definable function of positive integers). This definition is thought to be justified by considerations which follow, so far as positive justification can ever be obtained for the selection of a formal definition to correspond to intuitive notion.” [39]

Turing's definition:

“The ‘computable’ numbers include all numbers which would naturally be regarded as computable.” [169]

“Copeland’s entry is focused on the claim that the Church-Turing thesis was never meant to apply to machines. So he is adamant that we should never interpret Church or Turing as stating Thesis M. . . . This is nonsense. Church was enthusiastically entertaining some form of Thesis M.” [83]

A second, different reading of the thesis, which is again pointed out by Copeland, is the following.

**Thesis 5.6.3 (Thesis S)** *Any process that can be given a mathematical description (or that is scientifically describable or scientifically explicable) can be simulated by a Turing machine.*

Even in this case, it does not (or at least, would not) correspond to the original thesis.

“... any device or organ whose mathematical description involves functions that are not effectively calculable cannot be so simulated. As Turing showed, there are uncountably many such functions.” [42]

D. Goldin and P. Wegner [75], propose another interpretation. They believe that computation is a wider concept than function computation. With this premise they claim that the CT-T concerns only function computation, but it is wrongly applied to computation in general.

We think that the simple fact the CT-thesis needs an interpretation is relevant. Moreover, its “status” is not clear. Murawski and Wolenski [123] distinguish four lines about the status of the Church-Turing Thesis: (1) it is an empirical hypothesis, (2) it is an axiom or theorem (3) it is a definition, (4) it is an explication. Some authors suggest a possible refutation.

“We have two possible ways for refutation. One possibility for refutation is to eliminate equivalence in one direction, it means: If a function  $f$  is Turing computable do not imply that function  $f$  is effectively calculable . . .

Another possibility for refutation is to eliminate the equivalence in the other direction, it means: If a function  $f$  is effectively calculable do not imply that function  $f$  is Turing computable. [161]

Some authors suggest a possible proof.

“[It] may seem that it is impossible to give a proof of Church’s Thesis. However, this is not necessarily the case. . . . In other words, we can write down some axioms about computable functions which most people would agree are evidently true. It might be possible to prove Church’s Thesis from such axioms” [160]

For example, in [49], we can find an axiomatization and a proof. We are not interested in establishing if proofs of the thesis are correct or not. From our point of view, it is sufficient to stress the different opinions that we can find in literature about the necessity (or not necessity) of a proof. In fact, not only the CT-T is differently interpreted and often misunderstood, but also there is not a total agreement about the necessity of proving it. Some authors think it is possible, some others that it is meaningless since it is not a formal statement. Unusually, computer scientists do not concur if it is a definition, an empirical hypothesis, an axiom or a theorem to be proven. The most common position is considering it true, even without a proof. Its justification is based on its persuasiveness (which is huge). All these elements make the Church-Turing Thesis look like an epistemological assumption. This is remarkable, in our opinion.

## 5.7 Teleological approach to scientific knowledge

In this section we focus on the concepts of goals and objectives, which pervade this thesis. In particular, we dare a possible (audacious) link. The concepts of goal and requirement, used in systems design, have their conceptual “ancestors” in the Aristotelian *final causes*.

For empiricists, the concept itself of teleological explanation of phenomena, namely the existence of purposes and objectives in nature for the sake of which things are done, is inadmissible. This would confer to nature something like a “free will”, which is incompatible with the idea of nature as mechanism. However, Aristotle advanced aims as one of his famous four causes: *material*, *formal*, *efficient* and *final*.

*“Aristotle was deeply committed to investigating and explaining natural phenomena, which is reflected all through the surviving treatises on natural philosophy [...] What unites the questions explored in these natural treatises, [...] is that they are predominantly questions asking for the purpose of things, or, as Aristotle puts it, questions asking for - that for the sake of which -. According to Aristotle’s understanding of scientific knowledge, the answers to these specific why questions constitute teleological explanations [...]”* [104]

Final causes (or *telos*) differ from other ones from many points of view. The most evident difference is that “normally” causes happen before effects while in teleological explanations are the effects which occur first. In a causal explanation a first event  $E_1$  happens at time  $t_1$  and a second one  $E_2$  at time  $t_2$ . This is not a sufficient condition to state that  $E_1$  causes  $E_2$ , but it is a necessary one. In teleological explanation this temporal sequence is inverted. The  $E_1$  happens at time  $t_1$  to serve the second one  $E_2$  at time  $t_2$ , which is the cause.

*“Whereas in a typical causal explanation the earlier-in-time cause explains the later-in-time effect, in teleological explanations, as traditionally understood, the later-in-time effect (that is, the aim or purpose for which something happened) explains the earlier-in-time cause (that is, why something happened). The typical locution of a teleological explanation is: this happened in order that that should occur.” [138]*

Bacon recommended a limited use of final causes:

*“Bacon... quotes with approval the Aristotelien maxim - Vere scire est per causas scire - and the Aristotelien distinction of four causes, Materia, Forma, Efficiens, et Finis [but proposes ...] his famous condemnation of final causes [...] He blames their use in Physics; he approves their use in Metaphysics”. [170]*

Nevertheless, this kind of causes was admitted by authors such as Leibniz and Kant (among others).

*“ Leibniz did admit teleological explanations alongside mechanical ones. Apart from the need of teleological explanations (in terms of God’s purposes) in metaphysics, he argued that physical phenomena can be explained by mechanical as well as teleological principles. ... Indeed, Leibniz wholeheartedly accepted the Aristotelian final causes alongside efficient causes”. [136]*

The question is if science should admit or refuse final causes. We propose a compromise solution. In our opinion, the answer is that, anyway, they are actually used in everyday activity by engineers, during systems design, but are hidden by the use of a different terminology. Of course we do not claim the “airplanes want to fly” or “ships want to swim”. It would be an evident nonsense. However, stakeholders and systems have objectives, thus we simply suggest that the term “final causes” can have a (smooth) interpretation that is not incompatible with our *standard view* of science: the term “goal” is a (safe) synonym of the term “final cause”. From this point of view, we might say (quite provocatively), that requirements engineering and operations research are applied philosophy.

Part IV

**Conclusions**





# Chapter 6

## Conclusions

One of the main risks of systems design is producing something that is technically brilliant, but which does not fulfill the needs of the people who will use it or does not fit in the context where it will be used or, finally, does not respect the business model of its sponsor. From this point of view, there's no lack of examples. The Concorde aircraft [89], the Denver International Airport baggage system [156] and the Vajont Dam [47] are examples of systems that missed their global goal despite outstanding technical features because their operational context was not sufficiently considered (in certain cases with catastrophic consequences). It is emblematic that the term *Concorde syndrome* is used in project management “folklore” to refer to the obstinacy of persisting on choices that appear technically engaging with no regard for contextual bad feedbacks. Designers can quite easily make the mistake of focusing only on the technical features of the target system, looking for efficiency and performance while the most important thing is the effectiveness of the system, namely its capacity to answer to a need.

One of the claims of this work is that if the needs of the stakeholders, who are supposed to use the system-to-be, and the constraints of the environment, in which the system will work, are included as soon as possible in the design process, then the misunderstandings mentioned above can be reduced. In fact we use a systemic, i.e. holistic and top-down approach, inspired by Systems Engineering (SE) methods, that considers the system and its environment as a whole. Therefore, the first step of any project should be the definition of an abstract model, namely that considers only high-level system requirements. The set of stakeholders' needs should be as broad as possible to ensure a global vision, even if this means to deal with requirements that can be not technical and not-epistemic<sup>1</sup>. The drawback is that high-level requirements can be blurred and contradictory. We need methods to make them clear.

We think that formal methods can be used since the early design phases in order

---

<sup>1</sup>See 5.1.1.1, for a clarification of this term.

to clarify stakeholders' needs and system's goals and that Mathematical Programming (MP) (which is commonly used quite late in the design process, to optimize the constructional features of a system) is a viable technique from this point of view.

Fundamentally, we mix MP and SE modelling methods. We claim that this synergy can facilitate systems design. In order to show the validity of this join, we have provided three examples of application of high-level requirements integration in optimization problems with reference to three actual different systems. Three cases are not sufficient to *prove* a scientific claim, nevertheless the heterogeneity of the kinds of systems that we have considered endorses the generality of the applicability of our approach. Moreover, since our claim is *methodological*, this type of justification is legitimate (see Part 5 for a discussion on the different approaches to scientific knowledge).

1. The *information system architecture evolution management problem*, namely the problem of scheduling the replacement of existing services with new services without discontinuity.

The problem has considerable practical importance, but was never previously formalized to the extent we discussed. Our formalization, which is new, is the main contribution of the chapter.

The scenario is complex. The presence of many decision-makers, as business department managers, IT project managers and kill managers, entails several goals and we have provided an analysis of the underlying trade-offs. In fact different stakeholders have different needs that the evolution of the system has to satisfy and this causes conflicts between the respective tasks, especially when the scheduling of the activities is tight. The decision makers typically aim to gain: (1) top business value produced by the new services, (2) the maximum number of new useful modules activated and (3) the maximum number of useless modules deactivated. In most situations, the objectives (1) and (2) are not really conflicting since the activation of new services require new modules, thus Business and IT managers push the activities in the same direction. On the contrary, the objective (3) is potentially controversial, when there is a lack of time and resources. The activation of new modules and the deactivation of old ones requires work. If the amount of workforce is limited, as is usually the case, we then have to decide what has to be done first and, eventually, what is not necessary and can be planned for a later period. Business and project managers on one side and kill managers on the other have to compete for the existing resources and employ them for diverging aims. The former can fully attain their tasks on time only forcing the latter to delay theirs and vice versa. We have proposed a MP formulation that models the problem correctly, provide a theoretical analysis thereof, showing exactly where the

---

source of trade-off lies, and verify empirically that it can be used as a practical tool to solve realistically-structured instances.

2. The *recommendation problem*, namely the problem of providing reliable predictions of the ratings that a given user would express for given items, which is the basic function of a recommender system.

We have provided an abstract analysis of the global goals that a recommender system shall pursue and we have presented three different algorithms for the recommendation problem (TMW, BMC, LSPR). Each one is based on a different technique. TMW models the *word-of-mouth* in the social network of similar users as a flow, and exploits a pure MP approach. BMC adopts modularity based clustering methods for bipartite networks (which represents properly the typical input data that are used in this context). LSPR is based on information retrieval techniques since we guessed that retrieving interesting documents is a special case of retrieving interesting items and consequently we can transfer methods ideas from one field to the other.

A recommender system can help users by suggesting good items only if it can forecast correctly their potential ratings. In the case of RS design, a possible pitfall, due to a misunderstanding of the stakeholders' needs, consists in privileging computational efficiency to the detriment of effectiveness, namely the capability of providing good recommendations. Nevertheless, requiring, from the beginning, that the system shall be accurate and interesting reduces this risk. We formalize these goals and integrate them as criteria of a decision problem that can help the decision-maker to choose the best recommender system among a set of available ones in order to maximize users' satisfaction. We use our "home made" recommender systems as possible choices. Nevertheless the approach is general. This is the general contribution of the chapter. We have analyzed the trade-off between goals. A quite natural hypothesis, in the initial design phases, is that accuracy and quickness are in opposition. We have showed that this is actually the case. From this point of view, we exploit our approach to identify clearly which goals are in contrast to each other. More specific contributions are: the recommender systems themselves, which are brand-new or exploit methods never used before in RS fields (in particular TMW that has been used in a real application); their empirical evaluation; an experimental study of modularity based clustering and LSPR with RS dataset; a new simple metric, audacity, for RS testing.

3. The *equitable hazardous material transportation problem*, namely the problem arising in the transportation of hazardous material on a road network from one or many production points to one or many garbage dumps, crossing different areas.

This problem is well addressed in literature, but we have considered a particular type of constraint, namely equity. The seek of equity is a clear example of soft goal that is not easy to formalize. The main contribution of the chapter consists in the formal formulation of two definitions of equity and their integration in a MP model. The first is a form of straightforward egalitarianism, as it asks for similar values of risk for all the interested areas. The second one, more sophisticated from a philosophical point of view, admits an unequal distribution of available resources if this can help the most disadvantaged member of a group. In this context it means to plan the transportation in a way that reduces the risk for the most exposed area even if this increases the global risk. Our finding is that equity makes global risk increase, with both the interpretations introduced above. Rawls' s principle induces the higher increment of global risk. This shows that safety and equity are in opposition, in this context. The results is influenced by the features of the instances, but method and model are general.

Other contributions of the chapter concerns the computational efficiency of our approach. Optimization problems can be solved by simply “modelling them” using MP and calling an appropriate solver. This is called the *model-and-solve* approach to optimization (that we use extensively in this work). It adopts many features of a declarative programming paradigm since its control flow is not explicitly defined (by comparison with the “common” imperative paradigm). Notice that, many different formulations can describe the same optimization problem and the practical efficiency of solvers depends on both formulation and problem itself. Notice also that there is a linguistic, even rhetoric (in the noble sense of the term, as in [66]) dimension in this approach. The way problems are described influences their solution. In fact, well formulated models are more efficient than badly formulated ones, even if the underlying problem that we describe is the same (see Chapter 5 for more observations on this point).

We have applied some reformulations in order to make the MP hazmat model more efficient. The reformulations we adopt are exact. This means that we have improved the efficiency of our approach without compromising safety and equity.

We provide also a heuristic for accelerating the solution process through the reduction of the size of the instance, preserving its most fundamental feature. This approach is a kind of abstraction of the system domain, one of the technique that SE endorses.

## Part V

# Bibliography and appendices



# Bibliography

- [1] P. Abrahamsson, O. Salo, J. Ronkainen, and J. Warsta. Agile software development methods - review and analysis. Technical Report 478, VTT PUBLICATIONS, 2002.
- [2] R. L. Ackoff. Business ethics and the entrepreneur. *Journal of Business Venturing*, 2(3):185–191, 1987.
- [3] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.
- [4] V. Akgun, E. Erkut, and B. R. On finding dissimilar paths. *European Journal of Operational Research*, (121):232–246, 2002.
- [5] L. B. Alberti. *De Re Aedificatoria Libri X. Alamani, 1485; English translation by James Leoni of the Italian translation by Cosimo Bartoli, Firenze, 1550. (The) Ten Books of Architecture. Reprint of the edition of London 1755; Tiranti, 1955.* 1485.
- [6] P. T. Anthonie Meijers. General editors: Dov M. Gabbay and J. Woods, editors. *Handbook of the Philosophy of Science. Volume 9: Philosophy of Technology and Engineering Sciences.* Elsevier BV, 2009.
- [7] J. Anthonisse. The rush in a directed graph. Technical Report BN9/71, Stichting Mahtematisch Centrum, Amsterdam, 1971.
- [8] V. authors. Systems engineering fundamentals. Defense Acquisition University Press, 2001. Fort Belvoir, Virginia.
- [9] M. Balabanović and Y. Shoham. Fab: content-based, collaborative recommendation. *Commun. ACM*, 40(3):66–72, Mar. 1997.
- [10] M. J. Barber. Modularity and community detection in bipartite networks. *Phys. Rev. E*, 76(6):066102, 2007.



- 
- [11] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R. C. Martin, S. Mellor, K. Schwaber, J. Sutherland, and D. Thomas. Manifesto for agile software development, 2001.
- [12] N. J. Belkin and W. B. Croft. Information filtering and information retrieval: Two sides of the same coin? *CACM*, 35:29–38, 1992.
- [13] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter. Branching and bounds tightening techniques for non-convex MINLP. *Optimization Methods and Software*, 24(4):597–634, 2009.
- [14] P. Bernus, K. Mertins, and G. Schmidt. *Handbook on Architectures of Information Systems*. Springer, Berlin, 2006.
- [15] B. S. Blanchard and W. J. Fabrycky. *Systems engineering and analysis*. Prentice-Hall international series in industrial and systems engineering. Pearson Prentice Hall, Upper Saddle River, NJ, 4. ed edition, 2006.
- [16] S. Bliudze and D. Krob. Towards a functional formalism for modelling complex industrial systems. In P. Bourguine, F. Kepes, and M. Schoenauer, editors, *European Conference on Complex Systems*, 2005.
- [17] S. Bliudze and D. Krob. Towards a functional formalism for modelling complex industrial systems. *ComplexUs, special Issue : Complex Systems - European Conference 2005*, 2(3-4):163–176, 2006.
- [18] B. W. Boehm. *Characteristics of Software Quality*. TRW series of software. North-Holland, 1978.
- [19] B. W. Boehm. A spiral model of software development and enhancement. *Computer*, 21(5):61–72, May 1988.
- [20] K. E. Boulding. The ethics of rational decision. *Management Science*, 12, 1966.
- [21] U. Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25:163–177, 2001.
- [22] U. Brandes. On variants of shortest-path betweenness centrality and their generic computation. *SOCIAL NETWORKS*, 30(2), 2008.
- [23] U. Brandes, D. Dellinger, M. Gaertler, R. Görke, M. Hofer, Z. Nikoloski, and D. Wagner. On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, 20(2):172–188, 2008.
- [24] J. Brans. Ethics and decision. *European Journal of Operational Research*, 136(2):340–352, 2002.

- [25] J. Brans. The management of the future: Ethics in OR: Respect, multi-criteria management, happiness. *European Journal of Operational Research*, 153(2):466–467, 2004.
- [26] J.-P. Brans and G. Gallo. Ethics in OR/MS: past, present and future. *Annals OR*, 153(1):165–178, 2007.
- [27] J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In G. Cooper and S. Moral, editors, *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 43–52, San Francisco, 1998. Morgan Kaufmann.
- [28] J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. Technical Report MSR-TR-98-12, Microsoft Research, 1998.
- [29] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, and A. Perini. Tropos: An agent-oriented software development methodology. *Journal of Autonomous Agents and Multi-Agent Systems*, 2004.
- [30] D. Bridge, M. H. Göker, L. McGinty, and B. Smyth. Case-based recommender systems. *Knowledge Engineering Review*, 20(3):315–320, 2005.
- [31] M. Bunge. *Technology as Applied Science*. 1966.
- [32] R. Burke. Knowledge-based recommender systems, 2000.
- [33] R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, Nov. 2002.
- [34] D. Carmel, N. Zwerdling, I. Guy, S. Ofek-Koifman, N. Har’el, I. Ronen, E. Uziel, S. Yogev, and S. Chernov. Personalized social search based on the user’s social network. In *Proceeding of the 18th ACM conference on Information and knowledge management*, pages 1227–1236. ACM, 2009.
- [35] S. A. Carnes. Institutional issues affecting the transport of hazardous materials in the united states: Anticipating strategic management needs. *Journal of Hazardous Materials*, 13(3):257 – 277, 1986.
- [36] P. Carotenuto, S. Giordani, and S. Ricciardelli. Finding minimum and equitable risk routes for hazmat shipments. 34(5):1304–1327, 2007.
- [37] Y. Caseau. *Performance du système d’information – Analyse de la valeur, organisation et management (in French)*. Dunod, Paris, 2007.
- [38] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos. *Non-Functional Requirements in Software Engineering*. Kluwer Academic Publishers, 2000.

- [39] A. Church. An unsolvable problem of elementary number theory. *American Journal of Mathematics*, 58(2):345–363, April 1936.
- [40] C. W. Churchman. Operations research as a profession. *Management Science*, 17, 1970.
- [41] M. Connor and J. Herlocker. Clustering items for collaborative filtering. In *SIGIR-2001 Workshop on Recommender Systems*, New Orleans, Louisiana, 2001.
- [42] B. J. Copeland. The church-turing thesis. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Fall 2008 edition, 2008.
- [43] A. Costa and M. Melucci. An information retrieval model based on Discrete Fourier Transform. In H. Cunningham, A. Hanbury, and S. Rüger, editors, *Proceedings of the 1st Information Retrieval Facility Conference*, volume 6107 of *Lecture Notes in Computer Science*, pages 84–99, Vienna, 2010. Springer, Heidelberg.
- [44] A. Costa and F. Roda. Recommender systems by means of information retrieval. In *Proceedings of The International Conference on Web Intelligence, Mining and Semantics (WIMS)*. ACM, 2011.
- [45] A. Dardenne, A. van Lamsweerde, and S. Fickas. Goal-directed requirements acquisition. In *SCIENCE OF COMPUTER PROGRAMMING*, pages 3–50, 1993.
- [46] M. Degemmis, P. Lops, and G. Semeraro. A content-collaborative recommender that exploits wordnet-based user profiles for neighborhood formation. *User Modeling and User-Adapted Interaction*, 17:217–255, 2007. 10.1007/s11257-006-9023-4.
- [47] M. Delle Rose. Decision-making errors and socio-political disputes over the vajont dam disaster. *Disaster advances*, 5(3):144–152, 2012.
- [48] P. Dell’Olmo, M. Gentili, and A. Scozzari. On finding dissimilar pareto-optimal paths. *European Journal of Operational Research*, 162(1):70–82, 2005.
- [49] N. Dershowitz and Y. Gurevich. A natural axiomatization of computability and proof of church’s thesis. *Bulletin of Symbolic Logic*, 14(3):299–350, 2008.
- [50] M. Dorato. Epistemic and nonepistemic values in science. In *Science Values and Objectivity*, Pittsburgh-Konstanz Series in the Philosophy and History of Science, pages 52–77. University of Pittsburgh Press, 2004.
- [51] J. Driver. The history of utilitarianism. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Summer 2009 edition, 2009.

- [52] R. Déry, M. Landry, and C. Banville. Revisiting the issue of model validation in OR: An epistemological view. *European Journal of Operational Research*, 66(2):168 – 183, 1993.
- [53] P. Duhem. *La théorie physique: son objet, et sa structure*. Bibliothèque de philosophie expérimentale. M. Riviere & Cie., 1906.
- [54] P. Duhem. *Sōzein ta phainomena: essai sur la notion de théorie physique de Platon à Galilée*. A. Hermann, 1908.
- [55] M. Ehrgott. *Multicriteria Optimization*. Springer, New York, 2005.
- [56] E. Erkut and A. Ingolfsson. Transport risk models for hazardous materials: revisited. *Oper. Res. Lett.*, 33(1):81–89, 2005.
- [57] E. Erkut and V. Verter. Modeling of transport risk for hazardous materials. *Oper. Res.*, 46(5):625–642, May 1998.
- [58] S. Even, A. Itai, and A. Shamir. On the Complexity of Timetable and Multi-commodity Flow Problems. *SIAM Journal on Computing*, 5(4):691–703, 1976.
- [59] J. Figueira, S. Greco, and M. Ehrgott. *Multiple Criteria Decision Analysis: State of the Art Surveys*. Springer Verlag, Boston, Dordrecht, London, 2005.
- [60] C. M. Fonseca and P. J. Fleming. Multiobjective optimization and multiple constraint handling with evolutionary algorithms-part i: A unified formulation. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 28:26–37, 1998.
- [61] R. Fortet. Applications de l’algèbre de Boole en recherche opérationnelle. *Revue Française de Recherche Opérationnelle*, 4:17–26, 1960.
- [62] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75 – 174, 2010.
- [63] R. Fourer and D. Gay. *The AMPL Book*. Duxbury Press, Pacific Grove, 2002.
- [64] L. Freeman. A Set of Measures of Centrality Based on Betweenness. *Sociometry*, 40(1):35–41, Mar. 1977.
- [65] L. Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1979.
- [66] J. Freese. *Aristotle: the art of rhetoric*. ARISTOTLE, VOL 22. Harvard University Press, 1926.
- [67] G. Gallo. Operations research and ethics: Responsibility, sharing and cooperation. *European Journal of Operational Research*, 153(2):468–476, 2004.

- [68] R. Gandy. Church's Thesis and Principles for Mechanisms. In H. J. K. J. Barwise and K. Kunen, editors, *The Kleene Symposium*, volume 101, chapter Studies in Logic and the Foundations of Mathematics, pages 123–148. Elsevier, 1980.
- [69] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.
- [70] V. Giakoumakis, D. Krob, L. Liberti, and F. Roda. Optimal technological architecture evolutions of information systems. In M. A. et al., editor, *Complex Systems Design and Management*, pages 137–148, Berlin, 2010. Springer.
- [71] F. Giunchiglia, J. Mylopoulos, and A. Perini. The tropos software development methodology: Processes, models and diagrams. 2002.
- [72] J. Golbeck. Generating predictive movie recommendations from trust in social networks. In *Proceedings of the 4th international conference on Trust Management*, iTrust'06, pages 93–104, Berlin, Heidelberg, 2006. Springer-Verlag.
- [73] D. Goldberg, D. Nichols, B. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, December 1992.
- [74] B. Golden, M. Aiguier, and D. Krob. Modeling of complex systems ii: A minimalist and unified semantics for heterogeneous integrated systems. *Applied Mathematics and Computation*, 218(16):8039–8055, 2012.
- [75] D. Goldin and P. Wegner. The interactive nature of computing: Refuting the strong church—turing thesis. *Minds Mach.*, 18(1):17–38, Mar. 2008.
- [76] I. Guy, N. Zwerdling, D. Carmel, I. Ronen, E. Uziel, S. Yogev, and S. Ofek-Koifman. Personalized recommendation of social software items based on social relations. In *Proceedings of the third ACM conference on Recommender systems*, pages 53–60. ACM, 2009.
- [77] I. Guy, N. Zwerdling, I. Ronen, D. Carmel, and E. Uziel. Social media recommendation based on people and tags. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 194–201. ACM, 2010.
- [78] J. Habermas. Three normative models of democracy. In C. Polity Press, editor, *The Inclusion of the Other.*, pages 239–252.
- [79] J. Habermas. Discourse ethics, law and sittlichkeit. In P. Dews, editor, *Autonomy and Solidarity: Interviews with Jürgen Habermas*. Verso, 1992.

- [80] Y. Haimès, Lasdon, and L. D. Wismer. On a bicriterion formulation of the problems of integrated system identification and system optimization. *Transactions on Systems, Man, and Cybernetics*, (1):296–297, 1971.
- [81] N. Hanson. *Patterns of Discovery: An Inquiry Into the Conceptual Foundations of Science*. University Press, 1958.
- [82] J. L. Herlocker, J. A. Konstan, L. G. Terveen, John, and T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22:5–53, 2004.
- [83] A. Hodges. Alan turing in the stanford encyclopedia of philosophy.
- [84] A. Holder, editor. *Mathematical Programming Glossary*. INFORMS Computing Society, <http://glossary.computing.society.informs.org>, 2006–08. Originally authored by Harvey J. Greenberg, 1999-2006.
- [85] IBM. *ILOG CPLEX 12.2 User's Manual*. IBM, 2010.
- [86] P. Johnson, D. Joy, and D. Clarke. Highway 3.01, an enhancement routing model: Program, description, methodology and revised user's manual. Technical report, Oak Ridge National Laboratories, 1992.
- [87] H. Jonas. *Technik, Medizin und Ethik. Praxis des Prinzips Verantwortung*. Suhrkamp, Frankfurt am Main, 1996.
- [88] S. Kapurch. *NASA Systems Engineering Handbook*. DIANE Publishing Company, 2010.
- [89] N. Kelly. *The Concorde Story: 34 Years of Supersonic Air Travel*. Merchant Book, 2005.
- [90] S. Kleene. Recursive predicates and quantifiers. *Transactions of the American Mathematical Society*, 53:41–73, 1943.
- [91] S. Kleene. *Introduction to metamathematics*. Bibliotheca mathematica. North-Holland Pub. Co., 1952.
- [92] S. Kleene. *Mathematical Logic*. Wiley, New York, 1967.
- [93] M. Köksalan, J. Wallenius, and S. Zionts. *Multiple Criteria Decision Making: From Early History to the 21st Century*. World Scientific, 2011.
- [94] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl. Grouplens: applying collaborative filtering to usenet news. *Communication ACM*, 40(3):77–87, 1997.

- 
- [95] D. Krob. Modelling of complex software systems: A reasoned overview. In *FORTE*, pages 1–22, 2006.
- [96] M. Kuby, Z. Y. Xu, and X. D. Xie. A minimax method for finding the k best “differentiated” paths. *Geographical Analysis*, (29(4)):298–313, 1997.
- [97] T. Kuipers. Laws, theories and research programmes. In T. A. Kuipers, editor, *General Philosophy of Science: Focal Issues*, Handbook of the Philosophy of Science, chapter 1. Elsevier/North Holland.
- [98] T. A. Kuipers. *General Philosophy of Science: Focal Issues*. Handbook of the Philosophy of Science. Elsevier/North Holland, 2007.
- [99] M. Landry and M. Oral. In search of a valid view of model validation for operations research. *European Journal of Operational Research*, 66(2):161 – 167, 1993.
- [100] L. Laudan. *Progress and Its Problems: Towards a Theory of Scientific Growth*. Philosophy of science. University of California Press, 1978.
- [101] M. Le Menestrel and L. Van Wassenhove. Ethics outside, within, or beyond or models? *European Journal of Operational Research*, 153(2):477–484, 2004.
- [102] J. Lee, M. Sun, and G. Lebanon. A comparative study of collaborative filtering algorithms. *CoRR*, abs/1205.3193, 2012.
- [103] E. A. Leicht and M. E. J. Newman. Community structure in directed networks. *Phys. Rev. Lett.*, 100(11):118703, 2008.
- [104] M. Leunissen. *Explanation and Teleology in Aristotle’s Science of Nature*. Cambridge University Press, 2010.
- [105] L. Liberti. Mathematical programming: modelling and software. Lecture notes.
- [106] L. Liberti. Reformulations in mathematical programming: Definitions. In *CTW08 Proceedings*, pages 66–70, New York, NY, USA, 2008. G. Righini (ed.).
- [107] L. Liberti. Reformulations in mathematical programming: Definitions and systematics. *RAIRO-RO*, 43(1):55–86, 2009.
- [108] L. Liberti, S. Cafieri, and F. Tarissan. Reformulations in mathematical programming: A computational approach. In A. Abraham, A.-E. Hassanien, P. Siarry, and A. Engelbrecht, editors, *Foundations of Computational Intelligence Vol. 3*, number 203 in Studies in Computational Intelligence, pages 153–234. Springer, Berlin, 2009.



- [109] G. Linden, B. Smith, and J. York. Industry report: Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Distributed Systems Online*, 4(1), 2003.
- [110] P. Lipton. Engineering and truth, philosophy of engineering, vol.1.
- [111] G. List and M. Abkowitz. Estimates of current hazardous material flow patterns. *Transportation Quarterly*, (40):483–502, 1986.
- [112] X. Liu and T. Murata. An efficient algorithm for optimizing bipartite modularity in bipartite networks. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 14(4):408–415, 2010.
- [113] K. Lombard and R. Church. The gateway shortest path problem: generation of alternative routes for a corridor location problem. *Geographical System*, (1):25–45, 1994.
- [114] P. Lops, M. Gemmis, and G. Semeraro. Content-based recommender systems: State of the art and trends. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 73–105. Springer US, 2011.
- [115] J. Luftman. *Competing in the Information Age*. Oxford University Press, Oxford, 2003.
- [116] C. D. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.
- [117] E. McMullin. Values in science. *PSA: Proceedings of the Biennial Meeting of the Philosophy of Science Association*, 1982:3–28, 1982.
- [118] S. McNee, J. Riedl, and J. A. Konstan. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In *ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 1097–1101, Montréal, Québec, Canada, 22/04/2006 2006. ACM, ACM.
- [119] R. Merton. *Science, technology & society in seventeenth century England*. H. Fertig, 1970.
- [120] K. Miettinen. *Nonlinear multiobjective optimization*. Kluwer Academic Publishers, Boston, 1999.
- [121] J. Mingers. Ethics and OR: Operationalising discourse ethics. *European Journal of Operational Research*, 210(1):114 – 124, 2011.
- [122] G. Munda. Environmental economics, ecological economics, and the concept of sustainable development. *Environmental Values*, 6(2):213–233, May 1997.



- [123] R. Muraswski and J. Wolenski. The status of the church's thesis. In A. Olaszewski, J. Wolenski, and R. Janusz, editors, *Church's Thesis After 70 Years*. Ontos Verlag, illustrated edition edition, 2007.
- [124] J. Mylopoulos. The tropos project: A research overview, 2001. Tropos Workshop, Trento, November 15-16.
- [125] J. Mylopoulos, L. Chung, and B. Nixon. Representing and using nonfunctional requirements: A process-oriented approach. *IEEE Trans. Softw. Eng.*, 18(6):483–497, June 1992.
- [126] NASA, editor. *NASA Systems Engineering Handbook*. NASA, 1995.
- [127] M. E. J. Newman. Modularity and community structure in networks. *Proc. Natl. Acad. Sci. USA*, 103(23):8577–8582, 2006.
- [128] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69:026113, 2004.
- [129] I. Niiniluoto. *Is Science Progressive?* Synthese Library. Springer, 1984.
- [130] B. Nuseibeh and S. Easterbrook. Requirements engineering: a roadmap. In *Proceedings of the Conference on The Future of Software Engineering, ICSE '00*, pages 35–46, New York, NY, USA, 2000. ACM.
- [131] I. C. on Systems Engineering, editor. *INCOSE Systems Engineering Handbook*, volume 2.0. 2000.
- [132] M. Oztürk, A. Tsoukiàs, and P. Vincke. Preference modelling. In J. Figueira, S. Greco, and M. Ehrgott, editors, *Multiple Criteria Decision Analysis: State of the Art Surveys*, pages 27–72. Springer Verlag, Boston, Dordrecht, London, 2005.
- [133] M. Padberg. The boolean quadric polytope: some characteristics, facets and relatives. *Mathematical Programming*, 45:139–172, 1989.
- [134] C. M. Papadimitriou. *Computational complexity*. Addison-Wesley, Reading, Massachusetts, 1994.
- [135] K. Popper. *Conjectures and Refutations: The Growth of Scientific Knowledge*. Routledge Classics. Taylor & Francis, 2002.
- [136] K. Popper. Normal Science and its Dangers. In I. Lakatos and A. Musgrave, editors, *Criticism and the Growth of Knowledge*, pages 51–58. Cambridge University Press, New York, 2004.
- [137] S. Psillos. *Causation and explanation*. Central problems of philosophy. Acumen, 2002.

- [138] S. Psillos. *Past and contemporary perspectives on explanation in General Philosophy of Science: Focal Issues, vol.1*. Handbook of the Philosophy of Science. Elsevier/North Holland, 2007.
- [139] A. Punnen. A linear time algorithm for the maximum capacity path problem. *European Journal of Operational Research*, 53:402–404, 1991.
- [140] R. B. L. R. P. Feynman and M. Sands. *The Feynman Lectures on Physics*. Reading, MA: Addison-Wesley Publishing Company, 1963.
- [141] F. Raimondi. *Model Checking Multi-Agent Systems*. PhD thesis, University College London, 2006.
- [142] T. Ralphps. Introduction to mathematical programming. Lecture notes.
- [143] J. Rawls. *A Theory of Justice*. Belknap Press of Harvard University Press, Cambridge, Massachusetts, 1 edition, 1971.
- [144] J. Rawls. *Political Liberalism*. Harvard University Press, Cambridge, Massachusetts, 1993.
- [145] J. Rawls. *The Laws of Peoples*. Harvard University Press, Cambridge, Massachusetts, 1999.
- [146] J. Rawls and E. ed Kelly. *Justice As Fairness: A Restatement*. Biblioteca universitaria. Belknap Press, 2001.
- [147] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 Conference on Computer Supported Collaborative Work*, pages 175–186, 1994.
- [148] F. Roda and A. Costa. Bipartite modularity based clustering for collaborative filtering. In *Proceedings of the 11th Cologne-Twente workshop (CTW12) on Graphs and Combinatorial Optimization*, Munich, 2012.
- [149] F. Roda, L. Liberti, and F. Raimondi. Combinatorial optimization based recommender systems. In S. Cafieri, A. Mucherino, G. Nannicini, F. Tarissan, and L. Liberti, editors, *Proceedings of the 8<sup>th</sup> Cologne-Twente Workshop on Graphs and Combinatorial Optimization*, pages 175–179, Paris, 2009. École Polytechnique.
- [150] W. W. Royce. Managing the development of large software systems: concepts and techniques. In *Proc. IEEE WESTCON*. IEEE Press, August 1970. Reprinted in *Proc. Int’l Conf. Software Engineering (ICSE) 1989*, ACM Press, pp. 328-338.

- [151] A. Sage and J. Armstrong. *Introduction to systems engineering*. Wiley series in systems engineering. Wiley, 2000.
- [152] A. Sage and W. Rouse. *Handbook of systems engineering and management*. Wiley series in systems engineering. J. Wiley & Sons, 1999.
- [153] N. Sahinidis and M. Tawarmalani. *BARON 7.2.5: Global Optimization of Mixed-Integer Nonlinear Programs*, User's Manual, 2005.
- [154] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proceedings of the Fifth International Conference on Computer and Information Technology*, 2002.
- [155] M. Schlick. The future of philosophy. In *Proceedings of the Seventh International Congress of Philosophy*, pages 112–116, London, 1931.
- [156] M. Schloh. Analysis of the denver international airport baggage system. Technical report, Computer Science Department, School of Engineering, California Polytechnic State University, 1996.
- [157] G. Semeraro, P. Lops, P. Basile, and M. de Gemmis. Knowledge infusion into content-based recommender systems. In *Proceedings of the third ACM conference on Recommender systems*, RecSys '09, pages 301–304, New York, NY, USA, 2009. ACM.
- [158] U. Shardanand and P. Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, Denver, Colorado, USA, May 1995.
- [159] J. T. Shipley. *The Origins of English Words: A Discursive Dictionary of Indo-European Roots*. Johns Hopkins University Press, 1984.
- [160] J. R. Shoenfield. *Recursion Theory*, volume 1 of *Lecture Notes In Logic*. Springer-Verlag, Heidelberg, New York, 1991.
- [161] A. Sicard-Ramírez and M. E. Vélez-Ruiz. The church-turing thesis. Universidad EAFIT; Medellín, Colombia, <http://www1.eafit.edu.co/asicard/publications-talks/drafts.html>, 1999.
- [162] H. Simon. Rational choice and the structure of the environment. *Psychological Review*, 63:129–138, 1956.
- [163] H. Simon. A behavioural model of rational choice. In H. Simon, editor, *Models of man: social and rational; mathematical essays on rational human behavior in a social setting*, pages 241–260. J. Wiley, New York, 1957.

- 
- [164] R. I. Soare. Computability and recursion. *Bulletin of Symbolic Logic*, 2(3):284–321, 1996.
- [165] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Adv. in Artif. Intell.*, 2009:4:2–4:2, Jan. 2009.
- [166] A. Töscher, M. Jahrer, and R. M. Bell. The bigchaos solution to the netflix grand prize, 2009.
- [167] A. Tsoukiàs. On the concept of decision aiding process: an operational perspective. *Annals OR*, 154(1):3–27, 2007.
- [168] A. Tsoukiàs. From decision theory to decision aiding methodology. *European Journal of Operational Research*, 187(1):138–161, 2008.
- [169] A. M. Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42:230–265, 1936.
- [170] G. E. Underhill. The use and abuse of final causes. 2008.
- [171] S. Wall. Perfectionism in moral and political philosophy. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Winter 2012 edition, 2012.
- [172] W. A. Wallace. *Ethics in Modeling*. Pergamon, 1994.
- [173] F. Wenstøp. Operations research and ethics: development trends 1966-2009. *International Transactions in Operational Research*, 17(4):413–426, 2010.
- [174] L. Wittgenstein. *Philosophical Investigations*. Basil Blackwell, Oxford, 1953.
- [175] L. Wolsey. *Integer Programming*. Wiley, New York, 1998.
- [176] G. R. Xue, C. Lin, Q. Yang, W. Xi, H. J. Zeng, Y. Yu, and Z. Chen. Scalable collaborative filtering using cluster-based smoothing. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 114–121. ACM, 2005.
- [177] E. S. K. Yu. Towards modeling and reasoning support for early-phase requirements engineering. In *Proceedings of the 3rd IEEE International Symposium on Requirements Engineering*, RE '97, pages 226–235, Washington, DC, USA, 1997. IEEE Computer Society.



# Appendices



## Publications

### List of publications

#### JOURNAL PAPERS

1. V. Giakoumakis, D. Krob, L. Liberti, and **F. Roda**. *Technological architecture evolutions of information systems: Trade-off and optimization*, Concurrent Engineering: R&A, 20(2):127-147, 2012, <http://dx.doi.org/10.1177/1063293X12447715>.

#### CONFERENCES (REFEREED)

1. **F. Roda** A. Costa *Bipartite Modularity based clustering for Collaborative Filtering*, in I. Perseil, K. Breitman, and M. Pouzet, editors, Proceedings of the 17th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS), Paris, France, IEEE Computer Society, pages 23-28, 2012.
2. **F. Roda** A. Costa *Bipartite Modularity based clustering for Collaborative Filtering*, Proceedings of the 11th Cologne-Twente Workshop (CTW) on Graphs and Combinatorial Optimization, Munchen, Germany, 2012.
3. **F. Roda** A. Costa and L. Liberti, *Applications of Systems Architecture concepts* (poster session), CSDM 2011, Paris, France, 2011.
4. **F. Roda**, P.Hansen and L. Liberti *The price of equity in the hazmat transportation problem*, in L. Adacher, M. Flamini, G. Leo, G. Nicosia, A. Pacifici, and V. Piccialli, editors, Proceedings of the 10th Cologne-Twente Workshop (CTW) on Graphs and Combinatorial Optimization, pages 235-238, 2011.
5. A. Costa and **F. Roda**, *Recommender Systems by means of Information Retrieval*, in R. Akerkar, editor. Proceedings of the International Conference on Web Intelligence, Mining and Semantics (WIMS), Sogndal, Norway, May 25 - 27, 2011, ACM, pag. 57, 2011.



6. **F. Roda**, A. Costa and L. Liberti, *Optimal Recommender Systems Blending*, in R. Akerkar, editor. Proceedings of the International Conference on Web Intelligence, Mining and Semantics (WIMS), Sogndal, Norway, May 25 - 27, 2011, ACM, pag. 60, 2011.
7. **F. Roda**, L. Liberti and F.Raimondi , *Evaluation of collaborative filtering algorithms using a small dataset*, in J. Cordeiro and J. Filipe, editors, Proceedings of the 7th International Conference on Web Information Systems and Technologies (WEBIST), Noordwijkerhout, The Netherlands, SciTePress, pages 603-606, 2011.
8. **F. Roda** and L. Liberti, *Systems Engineering and Operations Research methods for transportation problems: a mixed approach* (extended abstract), ROADEF 2011 Proceedings, Saint-Etienne, France, 2011.
9. V. Giakoumakis, D. Krob, L. Liberti and **F. Roda**, *Optimal technological architecture evolutions of Information Systems*, in In M. Aiguier, F. Bretaudeau, and D. Krob, editors, Proceedings of the Conference Complex Systems Design & Management (CSDM) Paris, France, Springer, pages 137-148, 2010.
10. **F. Roda**, L. Liberti and F.Raimondi *Combinatorial optimization based recommender systems*, in S. Cafieri, A. Mucherino, G. Nannicini, F. Tarissan, and L. Liberti, editors, Proceedings of the 8th Cologne-Twente Workshop (CTW) on Graphs and Combinatorial Optimization, pages 175-179, 2009.