Designing for Cooperation - Incentive-Compatibility and Performance in Distributed Air Traffic Management Systems

Von der Fakultät für Ingenieurwissenschaften,

Abteilung Maschinenbau und Verfahrenstechnik der

Universität Duisburg-Essen

zur Erlangung des akademischen Grades

DOKTOR-INGENIEUR

genehmigte Dissertation

von

Hendrik Oliver Oberheid

aus

Mülheim an der Ruhr

Referent: Univ.-Prof. Dr.-Ing. Dirk Söffker
Korreferent: Univ.-Prof. Dr.-Ing. Dirk Kügler
Tag der mündlichen Prüfung: 5. September 2013

Contents

1. Introduction

The worldwide air traffic management (ATM) system is a distributed system, a network based on the collaboration of a large number of independent actors. The primary task of the ATM system is to provide a set of services which allow airspace users to safely and efficiently conduct flight operations. The main types of business organizations involved in that purpose are airlines, air navigation service providers, airports, and ground-handling services. In operational practice, each organization is involved through a number of individual sub-actors. These form the information- and decision-nodes within the ATM network. Where actors represent different companies or organizational units, they also often stand for different operational and economic interests.

The interaction of ATM actors follows elaborate rules, procedures, and protocols, with the purpose of establishing safe, efficient, and fair operations. Communication is performed both from human to human and through automated systems. Such systems and mechanisms frequently support e.g. traffic planning and allocation of airspace resources. The human-machine system as a whole can be interpreted (and modeled) as a multi-agent system. In such an abstract interpretation, the interaction of all agents finally produces the 'emergent behavior' of the ATM system. This emergent behavior determines both the efficiency of individual flights and the performance of the overall system.

Nowadays many of the procedures used in ATM can still be traced back to the beginnings of air traffic control (ATC). They often have their historical roots in the 1930s to 1950s when radio communication and radar control were established [Nola10]. Especially ground system development has been slow. Still typical of today are e.g. fixed decision responsibilities between actors, limited data exchange, inhomogeneous information availability in the air and on the ground, and a fragmentation of ground systems with disruptive information borders, especially in the European environment.

The plans for the future development and modernization of the ATM system are laid down for the EU in the ATM Masterplan [SESA09] and for the US in the NEXTGEN implementation plan [Fede09]. As part of these developments the rules and mechanism for the coordination between actors will become more and more sophisticated. Increasingly, elements of negotiation processes will appear in all operational areas and flight phases. Decisions will be based on real-time communication of operational preferences, constraints, and flight objectives. This development is subsumed under

the headings of collaborative decision making (CDM) and system-wide information management (SWIM).

The forces behind the development of CDM and SWIM concepts are of operational, economic and technological nature. Both aim at providing more capacity by a more efficient utilization of airspace resources, which is urgently needed. They also target efficiency and costs of individual flight operations. Technologically, more recent developments in communication, navigation, and surveillance (CNS) enable new ATM concepts which could not have been realized in the past. Whatever the reasons for specific new services may be, the central paradigm of the new ATM concepts in SESAR [SESA12] and NEXTGEN [Fede12] is more cooperation and intensified information sharing to enable improved ATM performance. The design and validation of new procedures and automated systems to support these developments will be a major engineering challenge in the coming years.

This thesis supports the view that improved coordination between all actors (i.e. stakeholders)[1] does indeed present an important potential to make ATM more efficient. Also, the stronger involvement of airspace users in all stages of the planning process seems the right and necessary step to take. However, CDM procedures do not automatically result in improved performance in practice, contrary to often overly optimistic expectations. Sometimes, the new procedures and mechanisms are in real life not able to produce the performance which was desired during system design. This is true despite all efforts to accommodate stakeholder interest and constraints better than in the past. Therefore, the following questions must be raised: What determines the success of these mechanisms? And how can potential failure due to misjudgement of behavior and performance be avoided?

In view of the questions put above, this work focuses on methods for modeling and predicting ATM-actors behavior in interaction with cooperative applications and mechanisms. It is based on two basic assumptions which are claimed to be necessary for the design of future ATM mechanisms and protocols. Both assumptions rely on the distinction between individual actors' objectives and overall system's objectives.

1. *Self-interested agents* Better provision of information and new options for negotiation will be used by agents to improve their individual situation and act in the interest of their organization. As long as certain rules (notably safety requirements) are obeyed, this behavior is intended, legitimate and an implication of a market-oriented system. However, the self-interested behavior does not always produce the choices that the system designer intended.

2. *Alignment of individual and system goals* Whether the local optimization of agents actually improves the overall emergent system performance depends on the resource allocation mechanism (i.e the operational mechanism that assigns ATM resources to agents, depending on their behavior, choices and situation). Local optimization of agents can either coincide or conflict with the interests of

[1] The term 'actor' is used synonymous with 'stakeholder' in the scope of this work.

other actors and overall system objectives for performance. Resource allocation rules and applicable procedures determine the alignment of actors' individual goals with the overall system goal.

An important research question thus seems to be how the alignment of individual goals and global system goals can be achieved through the systematic engineering of suitable mechanisms. Where actors have considerable freedom in their choices and where these choices affect performance issues, the understanding and the prediction of these decisions become an important engineering question.

1.1. Motivation

What is missing in today's ATM engineering approach to answer these questions raised above?

In the past decades the ATM community has established a set of principles for the best practice of system design and validation. This set is a foundation of methods which is based on and inspired by the generic disciplines of system engineering, software engineering, and requirements engineering.

The current approach to design and validation processes shows the following focus:

- *Design process:* The focus of the design process is currently *norm-oriented*. Typical design documents are textual documents that describe what the designer *expects* agents *should* do, in order to realize the system's (overall) goal and purpose. Typical examples of such normative descriptions are use cases, scenarios, and technical specification documents.
- *Validation & Verification (V&V) process:* The focus of the V&V process is *capability-oriented*.
  Typical validation plans e.g. define simulation exercises to find out
  - if the technical system *can* satisfy the requirements,
  - if human operators are *cognitively capable* of fulfilling their assumed roles,
  - if human operators *can interact* with the system at an *acceptable* workload level, situation awareness etc..

  Note that key performance measurements during the V&V phase are usually made under the implicit assumption that actors stick to their assigned roles and behave essentially as prescribed by the normative design. This means that actors are implicitly assumed to have the intention of doing what the system designer expects them to do and what they have been shown capable of doing. These assumptions may turn out not to be correct. The fulfillment of the capability-oriented requirements is necessary, but it is not sufficient.

Focus on actors' interests and systems' incentives

Thus, what is currently missing in the approach to validation and verification activities is a closer focus on the aspects of actors' interests, and - from the designer's point of view - possible incentives towards desirable actor behavior.

Definition 1.1.1 ***(Incentives)*** An incentive is any factor (e.g.. financial or operational) that provides a motive for a particular course of action, or counts as a reason for preferring one choice to the alternatives [Sull03].

With view to the expected developments of the ATM System in the areas of CDM and SWIM, the consideration of incentives should be gaining increasing importance. Essentially,

a) the informational situation is improving for all agents and

b) the influence of actors on decisions which were formerly centralized is increasing.

In consequence, it will become more and more important to analyze how actors can be expected to decide and how they will use these assets. Without this knowledge it will become increasingly difficult to make robust assumptions or predictions on the overall emergent system performance.

This leads to two research questions which motivate this work:

- How can the importance of incentives be reflected in the design of cooperative multi-agent ATM mechanisms?
- What kind of tools support are needed to design mechanisms which fulfill the criteria of incentive-compatibility?

1.2. Problem Statement

This thesis proposes the introduction of a new perspective on the design and validation process of ATM systems which is complementary to the existing normative-oriented and capability-oriented approach. In contrast to those, the new perspective puts agents' interests and system incentives in the center of attention.

To provide a tool for analysis from this perspective, this work develops a generic framework for the design and verification of incentive-compatibility in distributed systems. The framework defines a novel structure with an integrated modeling and analysis approach. The framework verifies the compliance of a system with a set of specific, formally defined requirements (in particular decision-theoretic and system-optimality criteria) and is thus termed a verification- rather than a validation tool.

The framework is then applied to the example problem of new potential cooperative arrival management process. This process is a highly relevant application as it is one of the central concepts currently under research in the major European and US-American ATM research programs SESAR and NEXTGEN.It presents a backbone of the operational ATM target concept to implement time-based navigation, and is therefore receiving considerable industrial- and research interest.

Modeling- and analysis-framework DAVIC

In order to practically incorporate the new incentive-compatibility perspective into the system engineering process, it must be adequately supported by methods, processes, and tools. For that purpose, this work develops a new framework called DAVIC, for the Design and Verification of Incentive-Compatible ATM Systems. A unique feature of the framework is that it combines

- network-based modeling and state space analysis, with
- decision theory and game theory

into an integrated modular structure. This structure makes it possible to analyze the compatibility of the incentives which are set in practice by the detailed system design of mechanisms with the designer's expectations.

Regarding the framework structure, DAVIC is built to allow an effective application in an engineering context through many reusable analysis criteria and algorithms. With the seamless integration of modeling, decision theory and optimization, it addresses a specific combination of problems in an engineering context. There, first the complexity of the mechanisms demands advanced modeling support to explore the solution space of the decision problem, and second the effects of candidate mechanisms should be open to analysis within an uninterrupted tool chain to support iterative development.

Although the framework is motivated by the ATM application context, it is not specific or limited to the ATM domain. The modeling and analysis methods used for the framework are well suited for the application to other kinds of distributed systems with a need for and interest in cooperation.

Framework application to arrival management

To demonstrate the framework and discuss its contribution and limitations, the approach is applied to an example problem which is arrival management. Arrival management is a service for organizing traffic in the vicinity of major airports. It includes subtasks such as sequencing and separating traffic during the approach and allocating runway resources to aircraft.

Arrival management procedures at large airports will see considerable changes in the next years, on the one hand driven by the demand for more arrival capacity with

more efficient (less noise, less fuel burn) approaches, and on the other hand enabled by technological advances in datalink technology. The trend is to have increasing elements of air-ground cooperation between aircraft and air traffic control, among other things to agree on precise overfly times for route points which fit both the ground-based and airborne needs. New planning mechanisms and protocols are being introduced as part of ground-based arrival management systems (AMANs) to manage sequencing and resource allocation tasks. Resource allocation of these mechanisms will be built upon information provided by aircraft.

Expected results

The thesis will provide results on two different levels:

On the *methodological level* results concern the DAVIC framework's capabilities and limitations. Here, this work discusses the applicability and general effectiveness of the modeling and analysis approach for the problem of incentive-compatible design in distributed systems. Benefits and added value of the framework (the contribution of DAVIC to the existing toolset) as well as problems and limitations are considered.

On the *application level* results concern the analysis of the specific arrival management problem and the considered planning algorithms. Here, the work discusses the consequences for the presented arrival management problem. The predicted behavior of aircraft is contrasted with desired behavior from the global system point-of-view. Finally, results and implications are discussed for the selection of the optimal arrival management mechanism from the design space.

1.3. Organization of the Work

This thesis is structured as follows:

The chapters 2-4 analyze the state of the art in the three research fields most relevant to the design of the target framework.

In chapter 2, the process and the application context of ATM system design and validation is studied. This leads to the proposal to add the analysis of incentives (and incentive- compatibility) as a new regular perspective during system design and validation. It also defines the need for new tools to support the analysis of agent's decisions in this context.

In chapter 3, decision-theoretic and game-theoretic background as well as system-optimality concepts are discussed. These are evaluated regarding their potential for the analysis and prediction of agents' decisions in distributed systems. However, in order to be applicable in a system engineering context to industrial scale problems, it is necessary to be able to derive appropriate representations of games from ATM-system models and mechanisms.

Chapter 4 focuses on the state of the art in Petri Net Modeling and related tools. The potential of Petri Nets for representing ATM mechanisms, computing the necessary game- and decision graphs, and analyzing such graphs with the concepts introduced in chapter 3 is discussed.

Chapter 5 presents the main methodological contribution of this work. It develops the framework 'DAVIC' for the Design and Verification of Incentive-Compatible systems. This chapter is independent of any specific application.

Chapter 6 applies the framework to a specific application example from ATM. It is used to optimize mechanisms for cooperative arrival management. This chapter fills each generic framework component with life.

Chapter 7 studies the simulation- & analysis-results from the application. Results are discussed both in terms of applied- and methodological consequences.

Chapter 8 summarizes the thesis and points out directions for future research.

2. ATM System Design and Validation

This chapter is concerned with the application background of this work in the Air Traffic Management (ATM) domain. It considers the state of the art regarding system design and validation within the ATM system. The discussion within this chapter is guided by the following questions:

- What constitutes the ATM system and what are its main characteristics (section 2.1)?
- What are important development trends in ATM and by which needs and influences are these developments driven. How are they related to cooperative behavior (section 2.2)?
- What are the current standards regarding design and validation methods in ATM, and what is in their focus (section 2.3)?
- How far are the current methods aligned with future development goals and with the needs of the ATM system to produce cooperative behavior (section 2.4)?

Given the complexity of the application domain, the presentation is necessarily selective and the answers to these questions are not exhaustive. The focus of the discussion is primarily on aspects of inter-actor cooperation and mechanisms for the coordination between actors in resource allocation problems.

In some instances, this chapter elaborates on aspects previously addressed in the introduction (chapter 1). It reviews those arguments considering a wider range of different sources and supplementary material.

The presented material shows that the current focus in ATM design and validation and the available methods only partly satisfy the development goals of ATM system designers. The detected shortcomings and the perceived misalignment of the method focus with the design problem motivate the development of the DAVIC framework which will be presented in chapter 5 of this work.

2.1. Air Traffic Management (ATM) System

In order to grasp important characteristics of the ATM system, the section starts with a review of key performance figures, technological challenges, and modernization efforts. This is supported with quantitative data. The section continues with a more

qualitative, system-theoretic view of the scope and structure of the ATM system. Finally, a distinction between airspace users' and service providers' perspectives is introduced.

Key figures

The air transport system is a complex sociotechnical system. It is also a system with major economic impact on a European and a worldwide scale.

According to a study on European air traffic [SESA06a], about 8.9 million commercial flights were counted over Europe in 2005. The flights were operated by close to 100 European airlines plus other international non-European ones. In total they carried approx. 650 million passengers and approx. 10.7 million tons of cargo.

By this commercial traffic, some 100 main European airports and more than 600 secondary airports were served. These airports were connected by some 600 airspace nodes operated by 36 different air navigation service providers (ANSPs) [SESA06a].

From a financial point of view, the direct stakeholders of the system accounted for an estimated 220 Billion Euros of added value, provided 4 Million direct or indirect jobs in the European economy and accounted for approx. 1.5.% of the European Gross Domestic Product (GDP).

The relationship between important ATM stakeholder categories, namely airspace users, air navigation service providers, airports, and supply industry is characterized in Figure 2.1. For each stakeholder category, key data to describe their role and situation are presented.

Growth- and capacity-challenge

The ATM system is facing major challenges if it is to cope with the predicted capacity demands of the future.

Over the past decade, European and worldwide air traffic has further grown despite some economic setbacks and growth is expected to continue. The number of flights in Europe was 10.5 million in 2010 according to [Euro11c] (approx. 29 million worldwide in 2007 [Offi07]). The Eurocontrol Long Term Forecast [Euro10] estimates a long term growth rate of 2.8% for Europe until the year 2030. This would mean that in 2030 there would be 1.8 times more flights than today.

Despite the positive economic effects, this growth is pushing many elements of the ATM to its technical performance limits. This is indicated by the continous delay situation in Europe. Despite intensive efforts to reach performance targets, air traffic flow management (ATFM) en-route delay for example has roughly doubled over the past decade [Euro11c]. It is widely accepted that today's ATM system is not up to facing the 2030 traffic expectations.

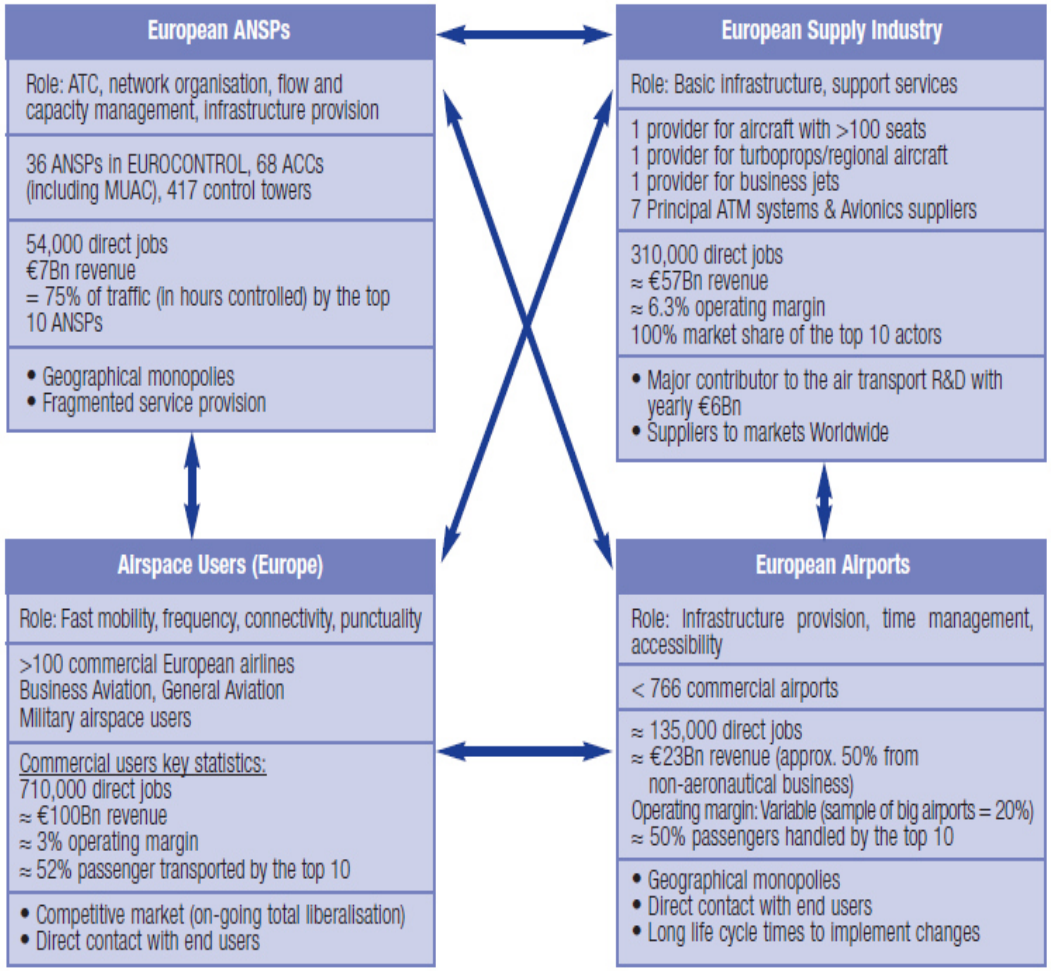

Figure 2.1.: Relationships forming the air transport value chain [SESA06a]

The capacity challenge is also due to the fact that many of the procedures (and technologies) still go back to the childhood days of air traffic control (ATC). ATM system development has traditionally been slow due to high safety and regulatory standards. Further, high investment costs and long life cycles of aircraft and ATC systems of often 20-30 years hinder rapid technological advancement.

Modernization programs

In order to cope with the challenges of modernization outlined above both Europe and the US are currently sponsoring major research efforts to develop systems, services and procedures for more productive ATM.

- For Europe a major share of resources is concentrated in the SESAR program (Single European Sky ATM Research program) [SESA12]. The program

presents research efforts with broad support, involvement and representation of all parts of the air and ground industry, ANSPs, airports, and airspace users.

- For the US a comparable program named NEXTGEN [Fede12] exists. The NEXTGEN program is a collaborative effort to transform the national airspace NAS, organized between the Federal Aviation Administration FAA and partners from the airlines, manufacturers, universities, associations, federal agencies, and the state.

Besides these two major programs, an important body of smaller, independent research projects obviously exists which have to be considered. Usually, the concepts dealt with inside the major programs can be traced back to previous smaller projects. The two major programs are used as indicators of current mainstream developments in ATM research and development. Due to the leading industrial role of the US and Europe in this area, significant technology impacts on a global scale can be expected. For any specific topic, additional sources have to be taken into account to consider the latest state of research.

System scope and structure

The following two definitions from ICAO Doc. 4444 [Inte01] describe Air Traffic Management (ATM) and the ATM system qualitatively. They provide a high-level definition of the application domain under consideration in this work.

Definition 2.1.1 ***Air Traffic Management(ATM)*** *The dynamic, integrated management of air traffic and airspace including air traffic services, airspace management and air traffic flow management — safely, economically, and efficiently — through the provision of facilities and seamless services in collaboration with all parties and involving airborne and ground-based functions.*

Definition 2.1.2 ***Air Traffic Management System*** *A system that provides ATM through the collaborative integration of humans, information, technology, facilities, and services, supported by air and ground- and/or space-based communications, navigation and surveillance.*

The following key aspects from these two definitions can be highlighted as being particularly relevant for this work:

- Both definitions stress the service-oriented characteristics of the system.
- The need for collaborative actions, cooperation, and coordination of many partners to provide useful services is underlined.
- A multitude of different technical and human actors are involved in every aspect of ATM. This renders the ATM system a prototypical multiagent system.

Note that it is common in literature on air traffic management to write both 'the ATM system' (with definite article) and 'an ATM system' (with indefinite article). Usually, the former refers to the overall, worldwide system facilitating air traffic, while the latter refers to a subsystem with more limited geographical, operational or technical scope. Moreover, 'an ATC-system' is usually thought to represent merely a technical system (excluding humans) which can form part of an ATM system.

Airspace users vs. service providers

In terms of the ATM value chain [SESA06a] some of the contributing agents primarily have a role as users of the ATM system and services. Others primarily are service providers. The users are generally airlines, military and general aviation traffic (GAT). The service providers primarily are the ANSPs (Air Navigation Service Providers) and airport actors but also ground-handling service providers. Service providers offer services which are needed by airspace users and they allocate certain resources to the users. The roles of users and service providers determine the key perspectives and business priorities. Note that this categorization reflects the dominant role, which however can be reversed for some special application contexts.

Airspace user perspective

The primary goal of the airspace user is to conduct a flight for business, private, recreational or military objectives. In order to do this, the user requires airspace resources and services and facilities of the ATM system to be provided in a coordinated manner. It is assumed that the user's flight objective may be expressed in terms of a user-preferred trajectory[1]. This user-preferred trajectory reflects the operation that the user would ideally like to conduct if the respective resources were available. Such a trajectory would include e.g. a flight route, vertical profile, and desired start times and speed (see [SESA06b]).[2]

Service provider perspective

The service provider perspective is characterized by the command over a certain set of resources and the responsibility and power to allocate these resources to airspace users. Key resources in the ATM system are airspace, en-route and airport slots, parking positions, sector capacity or runway time but also certain services (de-icing, push-back, fueling etc.).

[1] Also called 'business trajectory' in SESAR

[2] In today's operation the key data of this intent would be communicated to the ANSP in terms of an ICAO flight plan [Inte11]. In the future a more detailed sharing of intent will realized relying on element of the SESAR business trajectory (BDT) or Mission Trajectory (MDT).

Within certain regulative limits, the freedom of defining the resource allocation mechanism itself may reside with the service provider. However, the service provider will often also depend on the cooperation and information-sharing of its customers, the airspace users, to make optimal use of resources and maximize its own benefit where applicable.

Economic relation between actors

The ATM actors may conceive themselves either as competitors or as cooperation partners. This depends on a number of structural characteristics, regulation principles as well as the economic context and the specific operational situation (see also Figure 2.1).

Clearly today, the strongest competition exists between airlines (see [SESA06a]), due to their direct competition for the same market shares and customers. Direct competition also exist between some groundhandlers at the same airport if they offer the same services. On the other hand airports and ANSPs enjoy a kind of regional monopolies. This is so because they offer their services in different locations and cannot so easily be interchanged for each other only on the basis of the most economical pricing.

2.2. Increase in Cooperative System Elements

The future ATM system is expected to see a strong increase in cooperative system elements, coordination, and negotiation processes between ATM actors . In some instances, such processes have already become operational. For many other applications basic concepts exist, but the detailed mechanisms still have to be developed. The general trend however can be read from the so called 'operational target concepts' for the ATM system. These target concepts have been developed in cooperation by the ATM stakeholders to establish a common vision and development goal for the future system. The most relevant documents in that context are

- the Operational Concept Document of ICAO, the International Civil Aviation Authority [Inte05],
- the ATM target concept of SESAR for Europe [SESA06b], and
- the NextGen Concept of Operations for the US [Join07].

For their level and region of responsibility, these concepts describe the plans for the ATM system development. A time horizon of 10 to 30 years is considered.

Two high-level trends stated in the three concepts are highlighted here, as they are particularly relevant for this work (see e.g. [Inte05] appendix A6, [SESA06b] p. 18, [Join07] sections 2.3 and 4):

1. An increase in collaborative decision-making (CDM) elements is expected for all kinds of ATM services, and
2. A development towards enhanced system-wide information management (SWIM) between all actors of the ATM system is planned.

The following paragraphs give a short introduction to the philosophy of collaborative decision making (CDM) and the concept of system-wide information management (SWIM). Also, related research on ATM services is introduced in which these topics play a central role due to the presence of strong cooperative elements.

Collaborative Decision Making (CDM)

Collaborative Decision Making (CDM) has been a major topic in the ATM research at least since the 1990s. An introduction and overview to CDM is provided e.g. in [Ball00].

In the context of the European EATMP Program (European Air Traffic Management Program), the 'Ad-hoc Expert Group on CDM' was founded in 1998. The group agreed on the following working definition for Collaborative Decision Making [Mart99a]:

Definition 2.2.1 ***Collaborative Decision Making (CDM)*** *refers to a philosophy aimed at improving flight operations through increased involvement of Aircraft Operators and of Airport Operations in the process of Air Traffic Management (ATM). This covers applications aiming to take into account the internal priorities of the Aircraft Operators or the Airport, before and during the flight, and development of Information Management systems and procedures in order to make full use of available data.*

CDM-philosophy Wherever this promises efficiency gains for the overall system, the CDM philosophy foresees [Mart99a]:

- To share and exchange information between all partners of the ATM system,
- To notify all partners regarding the operational constraints of other partners,
- That all partners shall be prepared and willing to react to the constraints of other partners,
- That the results of decisions of individual partners shall be shared with all other partners,
- That decisions are taken by the actor who is in the best position to make this choice (subsidiarity principle).

Classification scheme Given the scope of the CDM definition the term has been used for a wide range of applications with very different collaborative elements. To categorize and compare the type and extent of collaboration, [Mart99b] introduced a classification scheme with four different levels of CDM:

- *Level 1:* Distribution of existing information to additional actors.
- *Level 2:* Cooperation to improve the planning estimates that are available to all.
- *Level 3:* Modification of planning processes to consider preferences and constraints of other actors, potentially in negotiation.
- *Level 4:* Dynamic allocation of decision-making authority to the actor in the best position to make a decision.

Application areas Prominent application areas for CDM have so far been the following:

- *Airport Management:* Airport CDM is meant to improve the way aircraft operators, ground-handling agents, airport operations, ATC and Central Flow Management Unit (CFMU) work together at an operational level. The following specific applications have been discussed or are already in operation at some airports:
  - Departure slot allocation: Airline and CFMU agree together on a new departure time slot (e.g. [Brin11]).
  - Departure sequencing: Airport, ground-handlers, airlines, and ATC collaborate on the departure sequence (e.g. [Jong05]).
  - Gate management: Airlines, ATC, and airport agree on gate changes (e.g. [Ober05, Jonk07b]).
  - Flight cancellations: In case of capacity reductions, airlines, airport, and ATC agree together on flights to be canceled.
  - De-icing and abnormal weather handling: Airlines and ground-handlers together agree on a de-icing sequence (e.g. [Dela03, Nori07]).

    In SESAR, airport CDM applications are treated in WP 6 'Airport Operations', particularly in SWP 6.5 'Collaborative Airport Planning' and Project 6.5.4 'Airport Operations Center (APOC) Definition'.
- *Trajectory Management:* Trajectory management is meant to change the way in which ground- and airborne actors agree on a trajectory. The trajectory should come as close as operationally feasible to what the airspace user would wish to fly. Trajectory management applications under discussion are:

- Air Traffic Control (ATC) agrees on a new trajectory and profile with an aircraft (e.g. due to weather changes or conflicting traffic).([Gree97, Shan98])
- Air Traffic Control (ATC) and aircraft agree on a target time.
- Arrival Management planning to agree on sequence and descent profile (e.g. [Korn06, Prev03]).

  In SESAR, trajectory management aspects are treated in WP 4 and 5 from an operational point of view and in WP 10 from the technical perspective.

System-Wide Information Management (SWIM)

The creation and improvement of System-Wide Information Management (SWIM) will be an important technical enabler for CDM. System-Wide Information Management is meant to provide the technical architecture and information management infrastructure to enable CDM procedures. On a technical level, the task of SWIM is to ensure that all the information gets to recipients in time. This information then enables collaborative and cooperative decisions [SESA].

Contrary to what might be expected from the general advances in information technology, information sharing between ATM actors can still be difficult today e.g. due to the following obstacles:

- A wide variety of applications and communication protocols has developed over time for very specific purposes.
- Many point-to-point communications with proprietary protocols exist between specific actors.
- The standard communication means between aircraft and ground is still voice radio and only very limited digital datalink coverage is available.
- On the ground, radar control centers are not readily connected due to system fragmentation and interoperability issues.
- Some actors and systems are not connected at all.

The development of a SWIM architecture will in the future allow more actors to technically participate in cooperative services. In SESAR, SWIM topics are mainly covered in WP 14 'SWIM technical architecture'. Industrial standardization plays a central role in the SWIM development as well as the commitment of ATM actors to invest in extensive infrastructure and equipment.

2.3. System Design and Validation Approach

Standard lifecycle model

A standard life-cycle model has evolved for the system-design and validation process in the ATM domain. The model is widely used to exchange on the maturity of proposed solutions and typical problems to be addressed at each life-cycle stage. The mainstream model for Europe has been laid down in the European Operational Concept Validation Methodology (E-OCVM) [Euro11a, Euro11b]. It is currently mandatory for all EU co-financed research to refer to this E-OCVM model. The US Federal Aviation Administration FAA and Eurocontrol have also endorsed a very similar model [FAAE08]).

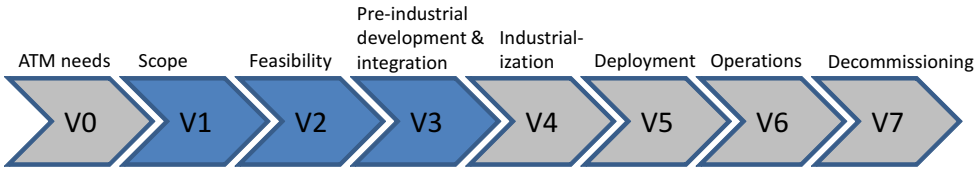

Figure 2.2.: Standard lifecycle model for ATM domain (adapted from [Euro11a])

In Figure 2.2 the ATM Concept lifecycle model of E-OCVM is depicted. The model consists of eight stages (V0-V7). These range from the initial elicitation of ATM needs over the development and operation of the system to its decommissioning. The most relevant phases from the point of view of research are V1: Scope, V2: Feasibility, and V3: Pre-industrial development & integration. From V4 onwards, the design, verification, and validation of new services and systems becomes increasingly stable. The technical systems are industrialized and put into operation.

V-Model system engineering approach

The process followed inside each of the individual stages V1 to V3 of the Life-cycle Model can be conceptualized in the form of a V-model system engineering approach [Broh99, Bund10] (see Figure 2.3) [3].

The V-model system Engineering Approach contains three types of activities:

1. *Decomposition:* On the left-hand decomposition arc high-level requirements from the overall operational concept documents (e.g. [Inte05] ,[SESA06b], [Join07]) are step by step broken down into domain-specific requirements and then into lower level technical specifications. This finally leads to a design of the system.

[3] Note that the unfavorable double use of the letter 'V' both for the labeling of the maturity steps (Figure 2.2) and for identifying the development model within a maturity step (see Figure 2.3) is coincidental. Care has to be taken to avoid confusion.

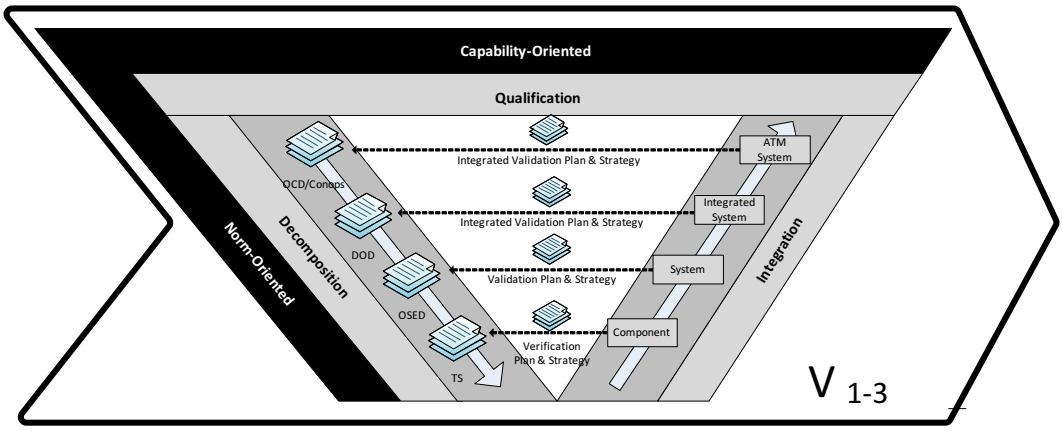

Figure 2.3.: Mapping standard design & qualification documents to V-model

2. *Integration:* On the right-hand integration arc, according to technical specifications components are constructed which are then assembled to subsystems and consequently more comprehensive integrated systems. Together with human operators, the fully integrated product forms a section of the overall ATM system.
3. *Qualification:* At each layer qualification tasks are carried out:
   a) *Verification* activities are performed at the lowest layer to show that components correctly implement their technical specification.
   b) *Validation* activities are performed at the higher layers to show that the integrated system satisfies the operational requirements described in the target concepts. Validation is often realized in human in the loop simulations or field trials.

In large development programs, it is common to support the individual steps with standardized document templates. These give some guidance regarding the information which is to be provided at each stage. In Figure 2.3 a mapping of common SESAR document templates to the V-model arcs is shown. These are used in the approx. 300 individual projects of the program.

Norm- and capability-orientation

The system engineering model in use and the priorities and constraints set by applicable guidelines and templates influence the focus of activities. For most of the work in ATM research the following observations are made (see Figure 2.3)

- *Design Phase (Decomposition Arc):*
  - The focus of this phase is norm-oriented.

  - The majority of the work focuses on (and is often limited to) describing a designer's normative expectation as to how the system (including humans!) should work.

- *Verification & Validation Phase (Qualification Arc) :*
  - The focus of this phase is capability-oriented.
  - The majority of the qualification work focuses on
    * proving that the system (both technical and human) is capable of acting according to its specification counterpart at the respective level of the V-Model decomposition arc, and
    * proving that the system is capable of realizing the expected operational benefits under the assumption that all agents act according to the norm.

In order to prove the expected benefits and exclude unfavourable side-effects, according to the European Operational Concept Validation Methodology [Euro11a, Euro11b], the following perspectives are recommended. The examination of each perspective should lead to the construction of cases with supporting material:

- *Business Case* - provides evidence for the realization of business objectives
- *Safety Case* - provides assurance of the achievement and maintenance of safety
- *Environment Case* - analyzes likely environmental impacts
- *Human Factors Case* - presents information on how the development affects humans in the system
- *Standards and Regulatory Case* - documents regulatory effects and constraints

2.4. Discussion of Current Approach

Alignment of design focus with system characteristics

In section 2.1 basic ATM system characteristics have been introduced, section 2.2 has reviewed current development trends and section 2.3 has discussed the prevalent focus of design and validation activities.

Considering the alignment of the curent mainstream engineering focus with the ATM system characteristics and design problems to be solved, a potential mismatch is observed. As illustrated in Figure 2.4, the misalignment becomes apparent from the contrasting of ATM system characteristics with development trends and with the employed engineering approach. The analysis provides the following results:

1. The *ATM system* is a complex, distributed, multi-agent system with heterogeneous roles. Within the system, relations between actors can be inherently

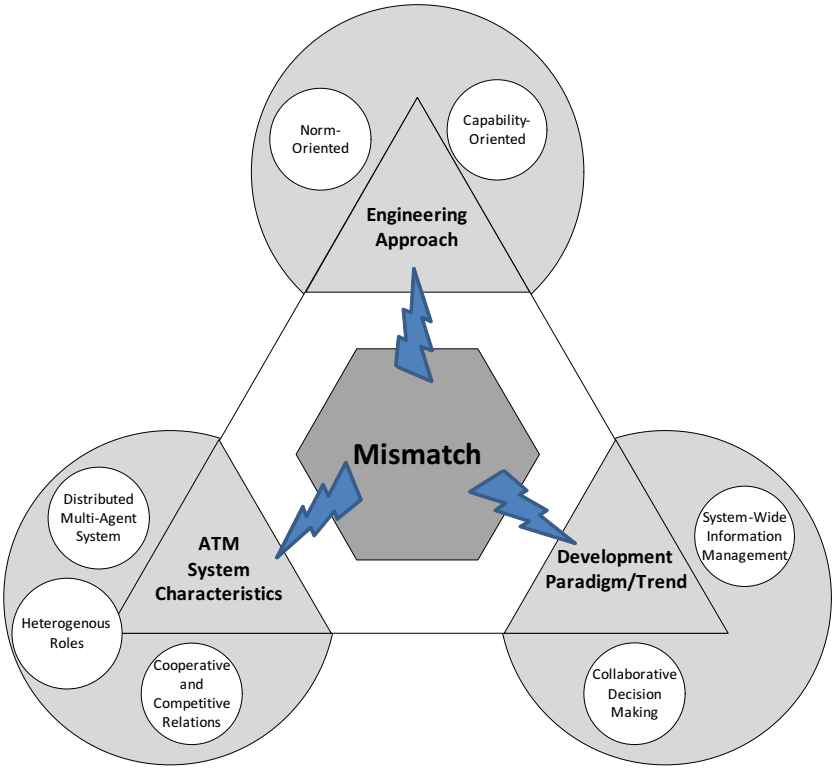

Figure 2.4.: Mismatch between ATM-system characteristics, development paradigm, and prevalent engineering approach

a) cooperative, but also

b) competitive.

2. The *ATM development paradigm* has the decentralization of decision-making and information sharing as important development goals. It includes the two key concepts

   a) Collaborative Decision Making (CDM), and

   b) System-Wide Information Management (SWIM)

3. The current *focus of the engineering process* for the mainstream of ATM work, is primarily

   a) Norm-oriented in the design phase, and

   b) Capability-oriented in the qualification activities (validation & verification).

In summary, the majority of ATM research today still assumes a cooperative attitude implicitly. This implicit assumption is also made where CDM and SWIM applications

with distributed decision-making- and negotiation elements are under discussion. A positive proof that cooperative behavior is actually in the interest of an individual self-interested agent is not regularly provided. The system design tends to formulate a norm of what actors *should do*. But capability-oriented qualification activities do not examine if compliance with this norm would be *rational*.

ATM research examining interests and incentives

As an exception to the rule of assuming cooperative behavior implicitely, the following research explicitly treats issues of interests and incentives within ATM-system design.

At this stage the discussion primarily concentrates on the ATM point of view. Some of the cited references will be referred to again for details in chapter 3 when the specific game-theoretic foundations have been introduced.

CDM context

- In [Ball00] a CDM overview is given and several CDM applications such as ground delay programs and collaborative routing are discussed for the US environment. Ball et al. recommend that game-theoretic approaches should be used to analyze the interdependent decision making in air transportation. Regarding cooperative behavior, they identify three basic forces that keep an airline from deviating from cooperative behavior: 1) goodwill towards CDM 2) peer pressure, to avoid negative reactions from others and 3) execution of supervision and enforcement functions by the Federal Aviation Administration FAA.
  In the long run, these three forces alone may not be strong enough, and other influencing factors may make that airlines could potentially gain from delaying information. Since most airline information is proprietary, this would be hard to detect. In that sense, the authors draw parallels to the prisoner's dilemma in game theory, where actors do not realize the best result by cooperating because they cannot rely on others to do so as well.
  Ball et al. argue that, in the long run, the cooperative paradigm would have to be supported by the provision of adequate economic incentives and mention auction-based procedures as an option for suitable applications. They clearly identify game theory as a technique to tackle that problem. This however is not applied within the scope of their paper itself.
- In a Eurocontrol report on CDM for airport Brussels Zaventem [Dela02] Delain et al. point to the fact that actors have different and competing objectives and that the best use of the available capacity will only be attained once quantifiable benefits for each actor are proven. Disincentives for cooperation have to be removed. The paper recommends that obstacles that currently prevent the

distribution of relevant information to other actors should be studied. It is pointed out that e.g. for some of the studied applications (improvement of Off-Block time for gate & turnaround management) there are disincentives for provision of information due to the existence of a 'double-penalty' mechanism. While the importance of incentive structures is recognized, Delain et al. do not recommend any particular technique for analyzing and adjusting incentives correctly. They do however request that incentives should be quantified.

- In [Jonk07b] Jonker et. al. sketch a future scenario in which arrival-, gate- and departure planning is done distributedly by airlines, airports, aircraft, and other parties. Their introductory section provides a detailed discussion of the potentials but also risks of CDM applications if mechanisms are not designed correctly. The paper states that "with the introduction of advanced planning concepts in the CDM framework, the possibility to 'cheat the system' will also be exploited" and that the new trends regarding cooperation, negotiation and CDM also introduce a fundamental problem: "The programs all assume a cooperative attitude of participants. However, it is not unthinkable that participants will show a competitive attitude rather than a cooperative. This is the case in current practice already. Often it can be commercially attractive for an airline to provide delay information or to cancel a slot as late as possible, to avoid slots from being used by other airlines".
  Regarding the potential consequences of a system design based on such false assumptions, Jonker et al. state: "If participants have a competitive attitude, a distributed planning system that relies on information sharing and shared responsibility will perform poorly. As each individual participant will try to maximize its own utility, it might withhold or manipulate information. Also, in joint decisions it will do its best to reach those decisions which are best for itself and ignore the interests of others."
  In terms of motivation, the arguments provided in [Jonk07b] are similar to the motivation of this work. The authors remark that "one should not assume a cooperative attitude among the participants, but provide explicit incentives for cooperative behavior". The paper then focuses on the study of a specific auctioning protocol and market mechanism based on spender-signed money. Variants of this application and protocol are also studied in [Jonk04, Jonk05b, Jonk05a, Jonk07a, Jonk08].

Market mechanisms and auctions

The work in [Nils03][Ball03][Le05][Wasl06] and [Rani09] discusses in different forms market mechanism and auctions for slot allocation and pricing at airports. Auctions are obviously an instrument for direct ATM actor influence on a desired result. This makes the relevance of incentive structures obvious. The allocation of airport slots is a special application however with primarily economic effects and to a lesser extent real-time operational impacts. Initial slot allocation takes place usually before the

day of operations. It concerns rather the planning stage than the execution phase of operational air traffic control.

- In [Nils03] the advantages of introducing market mechanisms are discussed primarily from an economic perspective. The main argument in favor of these mechanisms is to allow the airline industry better access to scarce airport resources on equal grounds. A market mechanism for slot pricing is discussed, which is however not formally modelled or analyzed.
- Ball et al. [Ball03] suggest the need for three types of market mechanisms: an auction of long-term leases of arrival and/or departure slots, a market that supports inter-airline exchange of long-term leases, and a near-real-time market that allows for the exchange of slots on a particular day of operation.
- The report [Le05] presents a case study of auction-based slot allocation at Hartsfield Atlanta international airport. In [Wasl06] a market mechanism for distributed computation of efficient and equitable solutions in air-traffic flow-management (ATFM) is proposed. The mechanism includes an airline cost function and the paper addresses the issue of airline incentives.
- A mechanism for ATFM slot allocation is also discussed in [Rani09] who also state that today there is a lack of incentives to efficiently use ATFM slots. It is claimed for the proposed mechanism that every actor will be better off when a solution according to the auction mechanism is achieved, so that all actors should be interested in participating.

Modeling, simulation, and human factors

- In a report on modeling tools as part of the SESAR definition phase [SESA08], an analysis of gaps and shortcomings of existing validation methodologies and modeling tools is conducted. As one outcome the need for increased usage of 'gaming exercises' is identified and so called 'war gaming techniques' are proposed. While this result recognizes in principle the vulnerability of CDM oriented system designs to uncooperative behavior, the concept of the 'gaming exercises' is based on human-in-the-loop simulations. These are aimed at gaining empirical evidence for likely actor behavior rather than conducting model-based, mathematical arguments on rational behavior. Thus, the idea of 'gaming exercises' is here closer to the field of behavioral decision theory (see chapter 3) than to game theory or mathematical decision theory.
- Related to a human factors study on shared information between controllers and pilots in datalink trajectory negotiation reported in [Farl99] Hansman notes in a follow up paper [Hans00] that increased information sharing may have the potential to induce unstable gaming behavior between agents. Also from a human factors point of view, the potential issues of undesirable gaming behavior should thus be considered.

Introduction of new incentive-perspective

This chapter has highlighted a potential neglect of the perspective of incentives and interest in the current approach to ATM system design. While there is a body of work which recognizes the importance of these issues, the overall situation may be characterized as follows:

- There are currently a number of contributions (see above paragraph CDM context) that reflect and discuss the importance of correct incentives for a wide range of ATM applications. However, these contributions are often limited to the identification and framing of the problem. The development and application of potential techniques as a solution to ensure correct adjustment of incentives remains out of scope. Nevertheless, this work makes an invaluable contribution to the increase of awareness and understanding of the problem.
- There are further contributions treating the examination of incentives or the assurance of certain game theoretic properties with a higher degree of detail. Usually this examination is performed for a very specific mechanism (e.g. auction protocol) which is introduced as part of that work. The results are often highly specialized in terms of the application. The potential for transferring and generalizing parts of the methodological approach to other kinds of ATM problems is often not discussed. Consequently, some difficulty remains for other ATM projects to judge the relevance and applicability for other services.
- A framework for the analysis of incentive-compatibility issues, which addresses the aspects of modeling and decision-theoretic analysis and which supports the re-usability for a wider range of systems has not been found.

Based on the analysis of this chapter, this work proposes to add the analysis of incentives (and incentive-compatibility) as a new regular perspective during system design, validation, and verification. The perspective should be considered alongside the already established views such as business case, safety case, environment case, human factors case, and regulatory case (see section 2.3).

In Figure 2.5 a simple development process is visualized, where all these views together define the activities and questions studied during the system design and validation phase. Obviously, not all perspectives are equally relevant or critical for all types of applications. This is true for the incentive-perspective in the same way as it applies to the other cases (e.g. safety or environment).

In summary however, a greater awareness for the incentive-perspective should be established. In addition, the potential of creating a more generic analysis framework should be discussed. This should go beyond the individual application and provide some guidance for re-use within different ATM projects. Chapter 5 of this thesis presents work towards the development of such an analysis framework.

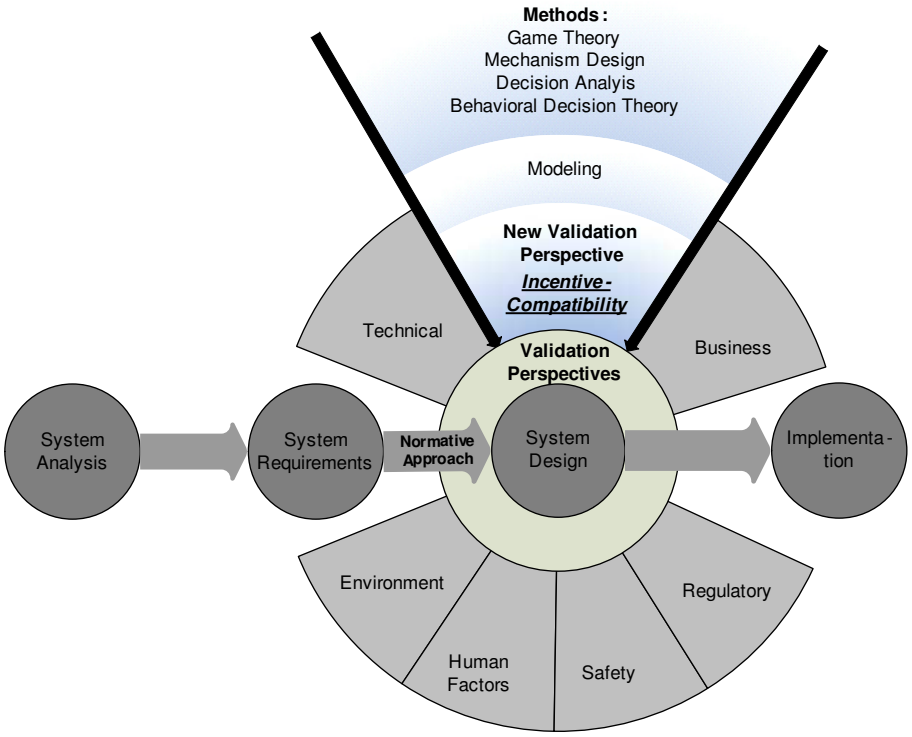

Figure 2.5.: Introducing incentive-compatibility as a new perspective to validation

3. Concepts of Individual Rationality and System Optimality

This section discusses the foundations of this work in the area of decision theory and system-optimality theory. As pointed out in chapters 1 and 2, the central approach of the framework which will be discussed in chapters 5 to 7 relies on contrasting

a) the *rational behavior* of self-interested agents (i.e. view of individual rationality)

with

b) the *desired behavior* of these agents from the point of view of the system designer (i.e. view of system optimality).

for given ATM mechanisms.

To realize (and model) the confrontation of these two perspectives, the exact meaning of *rational behavior* and *desired behavior* has to be formulated and formalized in mathematical terms.

The steps towards this formalization and related discussions in this chapter are organized as follows:

1. Section 3.1 provides a survey of available approaches to reasoning about agents decision-making and about *rationality*. It introduces decision analysis and game-theoretic concepts for the application in this work.
2. Section 3.2 introduces the mathematical formalizations of decision situations and of solution concepts which are needed to reason about *rational* decisions of agents.
3. Section 3.3 takes the designer's point of view: It formalizes the notion of *desirable* behavior and discusses criteria of system-optimality.
4. Section 3.4 summarizes the results and points out consequences for the application of agent-rationality and system-optimality concepts in the scope of this work.

3.1. Decision Theoretic Approaches

As formulated in [Hans05], Decision Theory (DT) is concerned with how subjects use their freedom. In situations and applications considered in decision theory there

are options to choose between and there is a subject (or various subjects) choosing between these options in a non-random way. Choices in theses situations are understood as goal-directed activities. Thus, decision theory is concerned with goal-directed behavior in the presence of options [Hans05].

Branches of decision theory

The field of decision theory is an interdisciplinary one, which has profited from contributions of researchers from such diverse disciplines as economics, statistics, psychology, political and social sciences, philosophy, mathematics, and engineering.

In these diverse disciplines, different streams and branches of decision science have developed. They promote different perspectives on - and questions towards - decision making. Four important branches of decision theory will be considered in the following:

- Decision Analysis (DA),
- Game Theory (GT),
- Behavioral Decision Theory (BDT), and
- Mechanism Design (MD).

Before these are discussed, it is useful to establish two conceptual dimensions in which decision-theoretic approaches are often characterized. These are (compare [Raif02]):

1. the number of actors taking part in the decision and included in the scope of the analysis, and
2. the perspective and orientation taken by the analyst towards the decision.

Dimension 1: Number of actors

For the first dimension, it is to be distinguished if

a) the decision situation (or game) is understood as an interaction of one individual/unitary player 'with nature', or

b) multiple actors with various, potentially interdependent strategies are taken into account.

In case a) nature is assumed to 'act' generally indifferently with regard to the human decision-maker's potential response strategies. The decision-making problem is mainly one of making probabilistic choices under uncertainty or risk.

In case b) several players are assumed to take into account that their individual outcomes depend not only on their own actions and risk/uncertainty, but also on the actions of other goal-directed players. The awareness and potential knowledge

about other players' preferences and possible actions can be used strategically in an argument of the following kind: 'I think that he will do, so I should ..but then he will think ..In which case I should ..' These considerations can lead to a (potentially infinite) circular regress in the anticipation of other people's strategies and choices.

Dimension 2: Analyst's pPerspective

For the second dimension, three main perspectives & orientations of decision theory are considered (see [Raif02]):

- Normative analysis - analyzing how ideally decisions should be made
- Descriptive analysis - describing how real people actually behave
- Prescriptive analysis - providing advice on how real people could in practice behave more advantageously

Meaning of 'should' in normative decision theory It is pointed out that the usage of 'should' in the sense of normative analysis can be interpreted in very different ways (see [Hans05]). However, there is broad agreement among decision theorists that in some way or the other the 'should' refers to the norms of *rational* decision-making. It is important to note that this is a very restrictive sense of the word *normative* which is much in contrast to its usage in other sciences, e.g. social sciences. It does not assume that rational decisions are by any means *desirable*.

Rational norm vs. social norm For the context of this work, the distinction between two kinds of norms is paramount:

- Normative analysis for the context of decision theory reflects a mathematical norm of what is considered rational behavior.
- The norm-oriented approach to system design described in chapter 2 in the first place reflects a social norm of desirable behavior which is subsequently translated to 'system-optimality concepts' in mathematical terms (see subsection 3.3).

Rational norm and social norm thus have distinct meanings. The approach of this work is to confront behavior according to these two norms. The goal is to achieve alignment between them.

3.1.1. Decision Analysis

The term 'Decision Analysis' (DA) goes back to work by R. Howard in 1964 [Howa66]. The type of problem concerned in decision analysis is the choice of an individual, unitary decision maker. This unitary decision maker seeks an optimal choice in the

interaction with *nature*. Typically the problem of making this choice is associated with conditions of uncertainty about the state and certain probabilistic events in the influence of nature–[Raif02][1].– Because of this, decision analysis is closely related to statistical decision theory, often adopting a Bayesian view to statistics (see e.g. [Smit88]).

The concept of 'interaction with nature' points to the fact that, while the probability of relevant events of nature may be a function of the agents' previous actions, nature does not make goal-directed decisions nor rationalize about the likely strategy of an agent to maximize its own benefit.

The advice offered by decision analysis regarding the decision situation is said to be 'prescriptive'. The intent of decision-analysis is to get from the normative claim of classical decision theory (to make rational decisions) to practical decision rules and solution concepts. These are meant to be applicable in practice to real-world decision problems. A decision rule or criterion is a specification of a procedure which may be used to identify the best alternative in any problem.

3.1.2. Game Theory

Game Theory (GT) is a mathematical theory of how rational actors should behave when their separate choices interact to produce payoffs to each player. Non-cooperative game theory, which is considered in this work, is about interactive, separate decision making. By definition, players make their decisions independently of each other (there is no collusion). These separate choices interact to determine the payoffs for each side [Luce89, Raif02].

The founding of modern game theory is usually attributed to the works of van Neumann [Neum28, Neum37] and van Neumann and Morgenstern [Neum44] with their work on the 'Theory of Games and Economic Behaviour'. In 1949 John Forbes Nash introduced the concept of equilibrium point [Nash50], which is today also known as 'Nash Equilibrium' (NE) (see section 3.2.3 for more details). This concept presented a milestone in the development of game theory.

In its basic form, game theory considers decision-making under certainty, in the sense that uncertainty is inherent only in other players' actions but no 'chance moves' of nature are considered. Additionally, extensions of game theory exist which treat probabilistic situations with incomplete information and uncertainty about other players' types and action sets and preferences (see e.g. [Myer04] and [Hars83] for work on Bayesian game theory).

[1] In contrast, the basic case of a decision problem of a unitary decision maker under certainty presents the classical optimization problem as treated in many engineering applications and operations-research e.g. through linear optimization.

Game theory vs. decision analysis

As Binsbergen [Bins07] points out, there is a number of commonalities between decision analysis and game theory as well as a number of important differences. The elements in game theory with direct counterparts in decision analysis are: (i) the strategy set, (ii) the moves of nature, (iii) the payoff mapping, (iv) the equilibrium concept. On the other hand decision analysis is different from game theory in that it lacks the following elements: (v) the other players (vi) the dependence of the payoffs on the actions of the other players.

Regarding the perspective taken by its analysis, game theory is said to be a *normative* theory. It is about how decision makers *should rationally* act although it has to be acknowledged that acting exactly to these norms may in practice not always be possible. Real-world actors may have relevant cognitive- and knowledge-limitations which hinder compliance with normative expectations.

Assumptions in game theory Beginning with the 1944 work by van Neumann and Morgenstern [Neum44], game theory has received important attention from outside mathematics. Especially economic sciences and social sciences have made extensive use of game-–theoretic concepts to explain or reason about conflict behavior of actors in their domain. However, the practical application of the normative theory has been subject to extensive scientific dispute. This is largely because game theory makes a number of strong assumptions about the games studied which are often difficult to satisfy in practice.

Common assumptions of game theory are (For assumptions 1-4 see [Raif02], for 5-8 see [Prin12]):

1. Players have to act (doing nothing is also understood as an act).
2. Players' payoffs depend both on what they do themselves and what other designated players do.
3. Players do not know what the others *will* do - but they know what the others *could* do.
4. Each player has a limited number of well-specified choices.
5. Every possible combination of plays leads to a well-defined end state.
6. A specified payoff for each player is associated with each end state.
7. Each player has perfect knowledge of the game and his opposition, that is he knows the full details of the rules of the game as well as the payoffs of all other players.
8. All players are rational, that is, each player, given two alternatives, will select the one that yields him the greater payoff (people take whatever actions are likely to make them more happy, and they know what makes them happy).

The last two assumptions in particular have been argued to restrict the application of game theory [Prin12] because they are regularly violated in real-world conflict situations. Work in behavioral decision-making (see subsection 3.1.3) has repeatedly shown that assumptions of unbounded rationality and unconditional maximization of payoffs don't usually hold for human behavior.

Further, for technical processes, knowledge of the full details of the rules cannot generally be assumed (see assumption 7). For example, in [Soff03] human interaction with technical systems is characterized as a game in which the rule base is only gradually established by the human operator through experience and learning.

Despite the difficulties in fulfilling some of its theoretical assumptions, game theory has enabled the analysis of many problems of interest in economics, management science and other fields [Prin12].

3.1.3. Behavioral Decision Theory

Behavioral Decision Theory (BDT) is about how real people actually make decisions in practice. Behavioral decision theory thus primarily adopts a *descriptive* approach to decision making–.

According to its aim of reflecting and explaining empirical findings about human decision making, basic research in behavioral decision theory is primarily a topic of cognitive psychology (compare [Raif02]). However, it has important applications of its findings e.g. in economics and in human factors engineering (see e.g. [Wick03]).

In the center of the behavioral analysis of decisions can be individuals or groups of decision makers.

Behavioral vs. normative approach Often the description of actual decision makers' behavior is presented with a normative model of game theory as a frame of reference for comparison (e.g. [Gint05]). That means that a special focus of behavioral decision theory is on investigating in which situations people act according to the expectations of a certain normative theory and where they do not.

The comparison with predictions from normative models has e.g. lead to the identification of characteristic biases, behavioral errors and anomalies in human decision making. Some pronounced deviations from the norms of unbounded rationality as proclaimed by normative theories have been identified [Came03].

An intensive argument has been carried out between supporters of behavioral vs. normative approaches to decision theory on the notion of 'rationality'. Among other things, this is related to the argument that formal normative models often do not take into account all relevant factors which are considered by human beings and thus cannot claim to determine rational behavior in an incompletely defined decision situation [Keys07]. Further, it has been shown that much of our actual decision-making behavior is actually driven by a desire to economize on how much time

individuals spend on making choices. While that doesn't always lead to optimal solutions for individual problems it may still be a rational approach in a wider context.

For certain types of decision situations (e.g. the prisoners' dilemma from game theory) human decision makers can actually come to more favorable overall outcomes for all participants than by acting according to the expectations of a normative model of rationality. Still, their behavior would be termed irrational under the axioms of traditional normative game theory.

3.1.4. Mechanism Design

Mechanism Design (MD) theory differs from the three branches of decision theory introduced above in the following sense:

- Decision analysis, game theory, and behavioral decision theory all assume the rules of the game as given and examine their rational and behavioral consequences.
- Mechanism design theory looks into the consequences of different types of rules primarily to construct a set of rules that achieves a specific kind of behavior.

Naturally, in its methodological basis, mechanism design relies heavily on the foundations of game theory and makes similar assumptions as introduced for game theory.

In the following, a number of key concepts of mechanism design are defined which are important within the scope of this work.

- *Incentive-Compatibility.* Hurwicz [Hurw72] introduced the ~~key~~ notion of 'incentive-compatibility', which allows the incentives of self-interested agents to be incorporated in the analysis. A mechanism is incentive-compatible if it is, according to a given solution concept, an equilibrium strategy (i.e. rational strategy, compare section 3.2.3) for each participating agent to report his information truthfully.
- *Strategy-Proofness.* A game is strategy-proof if there is no incentive for any player to lie about or hide private information from the other players. In other words, a mechanism is called strategy-proof if for every agent and for every possible bid (behavior) of other agents, the agent's utility is maximized when he tells the truth, i.e. truth-telling is a dominant strategy (see [Caro07]).
- *Impossibility- and Possibility Results.* A key focus of formal mechanism design lies on mathematical proofs if certain properties such as incentive-compatibility and strategy-proofness can be combined with other constraints in one mechanism (i.e. that the proof combination of the properties is 'possible'). A famous 'impossibility result' in mechanism design by Arrows (also know as Arrows impossibility theorem) has proven that for example no voting system can possibly meet a certain set of reasonable criteria (non-dictatorship, Pareto efficiency, and others) when there are three or more options to choose from [Arro51].

In summary, mechanism design studies how to implement good system-wide solutions to resource allocation problems that involve multiple self-interested actors considering each agent's private information and preferences (see [Park01]). Mechanism design provides a framework for analyzing a wide variety of institutions or resource allocation mechanisms with focus on the problems of private information and incentives [Roya07].

3.2. Formalization and Solution of Decision Problems

3.2.1. Games Against Nature

The basic problem of decision analysis is the individual decision of a unitary decision maker under uncertainty, also called a 'game against nature'.

Definition 3.2.1 The situation and decision problem of a ***Game Against Nature*** is defined as follows (see [Smit88]):

- The outcome space Θ describes all uncertainty in the control of nature.
- A single outcome is referred to as $\theta \in \Theta$.
- The decision space D describes all possible decisions for the agent.
- A single decision is referred to as $d \in D$.
- A utility function $U(d, \theta)$ specifies how much the agent will gain if he made a decision $d \in D$ and the outcome were $\theta \in \Theta$. The function $U(d, \theta)$ needs to be known for all values $d \in D$.
- For each decision $d \in D$ and all $\theta \in \Theta$ a probability mass function $p(\theta \mid d) \geqslant 0$ is defined with $\sum p(\theta \mid d) = 1$. The function $p(\theta \mid d)$ gives the probability that θ will take its various values given an actor choosing a decision $d \in D$.

Matrix representation The decision situation can be characterized by presenting $U(d, \theta)$ and $p(\theta \mid d)$ in matrix form (also called normal form) as in the following example. Rows present possible decisions d and columns present possible outcomes θ of nature.

$$
\begin{array}{rccccc}
 & & \theta_1 & \theta_2 & \theta_3 & \theta_4 \\
 & d_1 & 1 & 2 & 2 & 3 \\
U(d,\theta): & d_2 & 2 & 3 & 6 & 1 \\
 & d_3 & 6 & 6 & -3 & 0 \\
 & d_n & 3 & -2 & 3 & -1
\end{array}
\tag{3.1}
$$

$$p(\theta|d): \quad \begin{array}{c|cccc} & \theta_1 & \theta_2 & \theta_3 & \theta_4 \\ d_1 & 0.2 & 0.2 & 0.4 & 0.2 \\ d_2 & 0.8 & 0.1 & 0.1 & 0 \\ d_3 & 0.2 & 0.2 & 0.5 & 0.1 \\ d_n & 0.5 & 0 & 0.2 & 0.3 \end{array} \tag{3.2}$$

Solution concept In order to decide on an optimal decision, a solution concept is needed. A solution concept is a function that associates outcomes, or sets of outcomes, with decision situations or games [Auma87] and usually provides the payoffs that the outcome(s) yield to players. A solution concept can also be understood as a prediction of play, providing a mathematical argument which strategy should (in the normative/rational view) be adopted by the involved player(s) under a given set of assumptions.

Maximization of expected utility The standard solution concept for choosing an optimal decision to the decision problem formulated above is the maximization of expected utility. The concept is applicable to a risk-neutral decision maker. The expected utility of a decision is calculated by taking the weighted average of all possible utilities for that decision, with weights that reflect the probability that the respective outcomes will occur. Let Expected Utility EU be defined as

$$EU(d) = U(d, \theta_1) \cdot p(\theta_1|d) + U(d, \theta_2) \cdot p(\theta_2|d) + \ldots + U(d, \theta_n) \cdot p(\theta_n|d)\,. \tag{3.3}$$

As a result, the optimal decision d_{opt} is

$$d_{opt} \quad = \quad \arg\max_{d \in D} EU(d). \tag{3.4}$$

Alternative solution concepts for this decision problem are e.g. the 'MaxiMax' concept (aim for the best) or the 'MaxiMin' concept (avoid the worst), which apply to risk-taking and risk-averse decision makers.

3.2.2. Sequential Games Against Nature

Sequential games against nature describe multi-stage decision-analysis problems where agent decisions are interleaved with actions of nature. Still the problem is one of a unitary decision maker facing uncertainty.

Sequential games against nature are commonly represented as decision trees in 'extensive form' (see Figure 3.1).

Definition 3.2.2 The situation and decision problem of a ***Sequential Game Against Nature*** is defined by a directed graph with the following elements:

- A set N_D of *decision nodes* where a choice must be made (shown as squares).
- A set N_E of *event nodes* where uncertainty of nature is resolved (shown as circles).
- A set N_T of *terminal nodes* representing final outcomes for a combination of decisions and events (shown as bars).
- A set D of *decision branches* (mutually exclusive and collectively exhaustive), connecting decision nodes to event nodes.
- A set E *of event branches* (mutually exclusive and collectively exhaustive), connecting event nodes to decision nodes or terminal nodes.
- A (subjective) *probability* p, assigned to each event branch.
- A *utility* u, assigned to each terminal node.

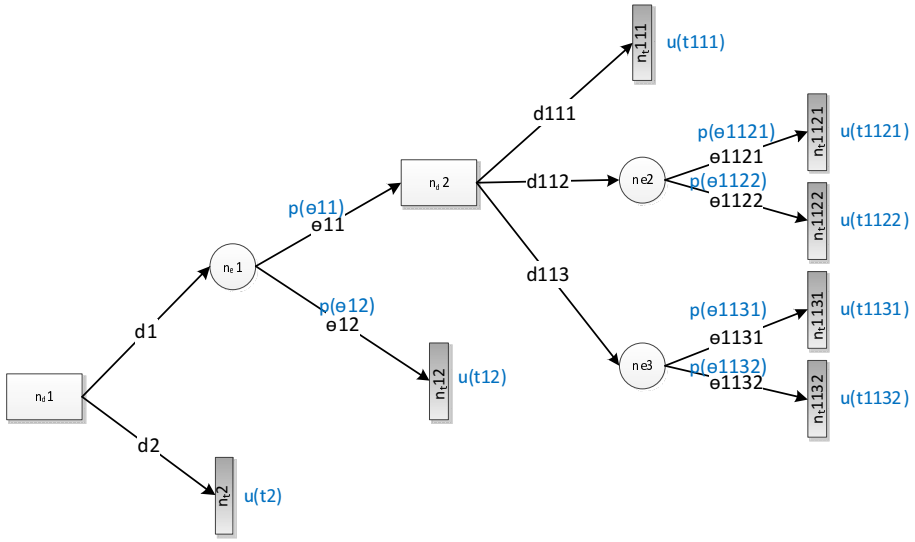

Figure 3.1.: Decision tree representing a sequential game against nature

Maximization of expected utility for sequential game In analogy to the one-stage decision problem, the optimal sequence of choices for the sequential decision problem can be determined by the maximization of expected utility. The solution relies on the backward induction method (also called roll-back method). Starting from the terminal nodes, utility values for each node are determined by the following rules:

1. For event nodes, the roll-back value is determined by the probability-weighted average of the successor node values.

2. For decision nodes, the rollback value is determined as the maximum of the successor node values.

After determination of all roll-back values, the optimal overall strategy is identified by going forward through the tree and choosing at each decision node the successor with optimal roll-back value.

As for the one-stage problem, e.g. the 'MaxiMax' concept (aim for the best) or the 'MaxiMin' concept (avoid the worst) may also be applied to the sequential game.

3.2.3. Simultaneous n-Player Games

With the introduction of the simultaneous n-player game, the step from decision analysis to game theory is made. In this game, n players make their choices simultaneously or at least do not observe other players' moves before making their own. Payoffs for each player are determined by the combination of actions.

In contrast to games against nature, interactions in this situation take place 'under certainty', in the sense that probabilistic influences of nature are not modeled. Typically, it is assumed that all agents know the payoffs and strategies available to other players, then the game is said to be one of 'complete information'.

The standard representation of this decision problem is the normal-form representation.

Definition 3.2.3 A ***Simultaneous n-Player Game*** in normal-form representation is described by the following data

- A finite set P of players is labeled $1, 2, ..., p$.
- Each player i in P has a finite number of pure strategies $S_i = 1, 2, ..., n_i$.
- A pure strategy profile is an association of strategies to players as a p-tuple $\Theta = (\theta_1, \theta_2, ..., \theta_p)$ with $\theta_1 \in S_1, \theta_2 \in S_2, ..., \theta_i \in S_i$.
- The set of strategy profiles is denoted by $\sum = S_1 \times ... \times S_p$.
- A utility function is defined as $U : \sum \rightarrow \mathbb{R}$.

The game in normal form is defined as a structure $\mathcal{G} = \langle P, \mathbf{S}, \mathrm{U} \rangle$ where $P = 1, 2, ...p$ is a set of players, $\mathbf{S} = (S_1, S_2, ..., S_p)$ is a~~n~~ p-tuple of pure strategy sets, and $\mathbf{U} = (U_1, U_2, ...U_p)$ is a~~n~~ p-tuple of payoff functions, one for each player.

Matrix representation Especially for 2-player games, normal-form games are often presented in matrix representation. The rows of the matrix denote potential strategies of player 1 and columns denote potential strategies of player 2. The entries in the fields of the matrix denote payoff pairs for all players for the respective strategy combinations.

		B	
		d1	*d2*
A	*d1*	0,0	-1,+1
	d2	+1,-1	-10,-10

For more than two players, a matrix representation can also be constructed by nesting of actors' choices but the representation loses some of its clarity.

Solution concepts (equilibrium concepts)

As for decision analysis, a suitable solution concept must be defined to associate an outcome (or set of outcomes) to an n-player game. In game theory, the solution concept is also called 'equilibrium concept'. Solutions are referred to as 'equilibria'. In analogy to physical systems, the behavior of players is expected to stay in equilibrium state (and therefore stable) until it is moved out of this stable state by external force. This stability relies on the fact that it is not advantageous for an individual player to unilaterally deviate from the strategy selection in the equilibrium.

The following solution concepts from game theory are relevant for this work:

Dominant Strategy equilibrium (DS)

The solution concept of dominant strategy equilibrium (DS) is based on the following two definitions:

Definition 3.2.4 ***Dominant strategy*** *A strategy is a dominant strategy if and only if it is a best response strategy, no matter what strategy the other agents select.*

Definition 3.2.5 ***Dominant strategy equilibrium*** *If, in a game, each player has a dominant strategy, and each player plays the dominant strategy, that combination of (dominant) strategies and the corresponding payoffs are said to constitute the dominant strategy equilibrium for that game.*

It is further distinguished between strict dominance and weak dominance.

Definition 3.2.6 ***Strict dominance*** *Strategy A* strictly dominates *B if it results always in better outcomes, regardless of what the other agents do. If A* strictly dominates *all other possible strategies, it is said to be* strictly dominant.

Definition 3.2.7 ***Weak dominance*** *Strategy A* weakly dominates *B if there is at least one set of opponents' actions where A is superior to B and it is at least as good for all other sets of opponents' actions. Strategy A is* weakly dominant *if it dominates all other strategies, but some are only* weakly dominated.

The dominant strategy equilibrium is a simple and robust solution concept, mainly because an agent doesn't need to form beliefs about either other agents' rationality or the distribution over the other agent types [Dash03]. However, the existence of a dominant strategy equilibrium is not guaranteed, i.e. for some games no solutions will be found.

Iterative Elimination of Dominated Strategies (IEDS)

For many games the players do not have dominant strategies and a dominant equilibrium cannot be identified directly. A solution based on dominance may nevertheless be feasible as the result of a procedure called 'Iterative Elimination of Dominated Strategies' (IEDS). The validity of this solution is based on the common knowledge of rationality.

Elimination process For the IEDS solution, dominated strategies are iteratively deleted from the game. The elimination is based on the assumption that it would be irrational to play a dominated strategy. Each elimination leads to a smaller game, in which often new strategies become dominated which were not dominated before. These are removed step by step. The process is continued until dominated strategies can no longer be found.

Definition 3.2.8 ***Dominant solvable*** *A game is called* dominant solvable *if only one strategy set remains.*

Two variants of IEDS exist, where either only strictly dominated strategies or strictly and weakly dominated strategies are eliminated:

- *Results of IEDS for strictly dominated strategies.* If after completion of the elimination process only one strategy combination remains, this is the valid solution and the unique Nash Equilibrium (see definition 3.2.9 for Nash Equilibrium below).
- *Results of IEDS for weakly dominated strategies.* When weakly dominated strategies are eliminated, the solution can depend on the order of elimination. If after completion of the elimination, a single strategy remains for each player, the strategy set is also a Nash Equilibrium. This may not be the only Nash Equilibrium though, and which NE is found can depend on the order of elimination.

Nash Equilibrium (NE)

A highly influental solution concept of game theory is the Nash Equilibrium [Nash50, Myer99].

Definition 3.2.9 ***Nash Equilibrium*** *A Nash Equilibrium, also called strategic equilibrium, is a list of strategies, one for each player, which has the property that no player can unilaterally change his strategy and thereby get a better payoff [Turo02].*

It is distinguished between pure strategy Nash Equilibria and mixed strategy Nash Equilibria.

Definition 3.2.10 ***Pure Strategy*** *A pure strategy defines a specific move or action that a player will follow in every possible attainable situation in a game. Such moves may not be random, or drawn from a distribution, as in the case of mixed strategies [Shor12].*

Definition 3.2.11 ***Mixed Strategy*** *A mixed strategy consists of several possible moves (pure strategies) and a probability distribution (collection of weights) which corresponds to how frequently each pure strategy is to be played [Shor12].*

Definition 3.2.12 ***Pure Nash Equilibrium*** *A pure strategy Nash equilibrium of a game is a Nash equilibrium in which all players use a pure strategy.*

Definition 3.2.13 ***Mixed Nash Equilibrium*** *A mixed strategy Nash equilibrium of a game is an equilibrium where at least one player uses a mixed strategy.*

Nash proved that every finite strategic-form game must have a Nash equilibrium, at least in mixed strategies:

Definition 3.2.14 ***Nash Existence*** *Every finite strategic-form game has a mixed-strategy Nash equilibrium [Nash50].*

Definition 3.2.15 ***Nash Counterexample*** *A strategy combination s is defined as a Nash Counterexample for s' if s proves that s' is not a NE. This means that s shows that one player can achieve a better payoff than in s' by unilateral deviation.*

What makes NE a natural solution concept is the fact that any prediction about the outcome of a non-cooperative game is self-defeating if it specifies an outcome that is not a NE. In [Nash50] Nash proved the existence of at least one NE (pure or mixed) for every game with a finite number of strategies for each player [Rapo00].

3.2.4. Sequential n-Player Games

Sequential n-Player games are games in which several actors decide one after another. A sequential game is called a game of perfect information if only one player moves at a time and each player knows every action of the players that moved before him at every point of the game. Sequential games are typically represented in so called extensive form.

Definition 3.2.16 A ***Sequential n-Player Game*** with perfect information in extensive form is defined by the following elements:

- A finite set P of players, labeled $1, 2, \ldots, p$
- A set of histories with typical member $h \in H$. h presents a sequence of actions by individual players. $\emptyset \in \mathbb{H}$ is the start of the game. If $h \in \mathbb{H}$ but there is no $(h, a) \in H$ where a is an action for some player, then h is 'terminal'. The set of terminal histories is denoted as $Z \subset H$.
- A function $F : H \setminus Z \longmapsto N$, assigning a player to each non-terminal history.
- Payoffs for each $i \in N$ are defined over terminal histories, $u_i : Z \longmapsto \mathbb{R}$

Any structure with these four features can be defined as an extensive-form game:

$$\mathcal{G} = \langle P, H, F, u_{ii \in N} \rangle.$$

Tree representation The graphical representation of an extensive-form game consists of a tree structure. The nodes of the tree represent all possible states of the game and the edges of the graph are moves. The initial node of the tree represents the first decision which is to be made. Every path of edges through the tree finally leads to a terminal node. An n-tuple of payoffs is assigned to each terminal node of the tree and defines the outcome for each player if the game ends at that node (see Figure 3.2). The player function defines a player for each of the non-terminal nodes of the tree.

Subgame Perfect Nash Equilibrium Nash Equilibrium is the central solution concept also for sequential games. Subgame Perfect Nash Equilibrium (SPE) is a refinement of the basic NE concept. The concept of SPE aims at eliminating non-credible strategies (non-credible threats) which would present NEs if choices were made simultaneously, but are no NEs of a subgame of the tree after the respective decision node within the tree has been reached. The following definitions are introduced:

Definition 3.2.17 ***Subgame*** *In a game of perfect information a subgame is a collection of nodes starting at a single node and containing all of its successor nodes.*

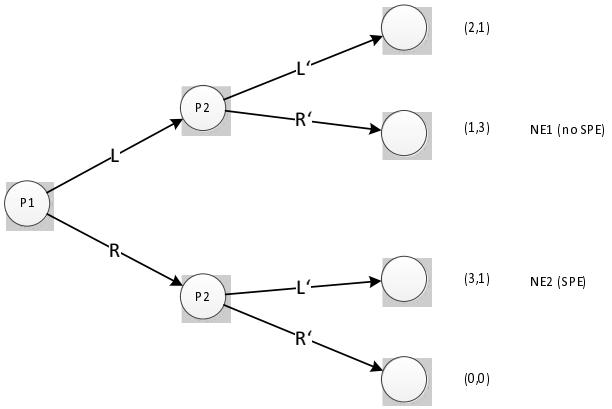

Figure 3.2.: Sequential 2-player game in extensive form (adapted from [Cane09])

Definition 3.2.18 ***Subgame Perfect Equilibrium*** *A Subgame Perfect Nash Equilibrium is an equilibrium such that players' strategies constitute a Nash Equilibrium in every subgame of the original game [Shor12].*

Finding SPEs Subgame Perfect Equilibria in games of perfect information can be found by backward induction. In a way similar to that for the maximization of expected utility in section 3.2.2, one works backwards through the tree from the terminal nodes to the decision nodes. First, the optimal choice of the player who makes the last move of the game is determined. Next, the last player's action is taken as given, and the optimal strategy of the next-to-last moving player is determined. The process continues in this way backwards through the tree until all players' actions have been determined [Shor12].

Example game Applying this process, the game in Figure 3.2 yields two NEs. NE1 is (L/R') and NE2 is (R/L'). However, only NE2 with (R/L') is a SPE and therefore credible. Taking NE1 as a starting point, it would be profitable for player 1 to deviate to R, since in a sequential game, after he has committed to R, player 2 would play L' to maximize his own payoff. The potential threat that player 2 could choose R' after player 1 has committed to R is not credible, since player 2 would also harm himself.

3.3. System-Optimality Concepts

System-optimality concepts denote criteria which state the designer's point of view on how the game shall be played in order to realize the intended system performance. System-optimality describes *desirable* choices (and outcomes) from the overall system perspective.

Within the area of system optimality, one important branch is based on agents' joint utility. Two such concepts, namely 'Hicks Optimality' and 'Pareto Optimality' are introduced in the following. They are application-independent in their formulation and are frequently used in conjunction with decision-theoretic and game-theoretic analysis.

The notion of system-optimality also includes criteria addressing specific aspects of agent behavior for certain applications or targeting application-specific system variables other than abstract utilities and preferences. In this context, examples of 'Agent Behavioral Measures' and 'Planning Quality Measures' will be introduced in later chapters in conjunction with the applications.

3.3.1. Hicks Optimality

Hicks optimality, named after the British economist John Hicks, describes outcomes for which the total payoff across all players is maximized. A Hicks-optimal outcome is always Pareto efficient (compare definition 3.3.4–). Naturally, each game must have at least one Hicks optimum.

Let sp be a strategy profile for p agents and J be the joint (summative) utility of all agents' individual utilities u for sp

$$J(sp) \quad = \quad \sum_{i=1}^{p} u(sp)_i \ . \tag{3.5}$$

Then Hicks optimality and the Hicks-optimal set are defined as follows:

Definition 3.3.1 ***Hicks optimality*** *Strategy profile sp is Hicks optimal if there does not exist another strategy profile $sp' \in SP$ for which the joint utility $J(sp') > J(sp)$.*

Definition 3.3.2 ***Hicks optimal set*** *The Hicks-optimal set S_H contains all strategy profiles $sp \in SP$ which are Hicks optimal.*

$$S_H \quad = \quad \arg\max_{sp \in SP} J(sp) \tag{3.6}$$

In the design of distributed systems, the maximization of the total payoff to all agents is obviously one important goal.

3.3.2. Pareto Optimality

Pareto optimality, named after the italian mathematician Vilfredo Pareto, describes an outcome which cannot be improved upon without hurting (i.e. giving a lower utility u to) at least one player p.

Pareto domination, Pareto optimality and the Pareto-optimal set are defined as follows:

Definition 3.3.3 ***Pareto domination*** *Strategy profile sp Pareto dominates strategy profile sp' if it is true for all players i that $u_i(sp) \geqslant u_i(sp')$, and there exists at least one player j for which $u_j(sp) > u_j(sp')$ [Shoh08].*

Definition 3.3.4 ***Pareto optimality*** *Strategy profile sp is Pareto optimal, or strictly Pareto efficient, if there does not exist another strategy profile $sp' \in SP$ that Pareto-dominates s [Shoh08].*

Definition 3.3.5 ***Pareto-optimal set*** *The set S_P of all strategy profiles $sp \in SP$ which are Pareto optimal, such that*

$$S_P \; : \; = \; \left\{ sp \in SP \mid \neg \exists sp' \in SP \, sp' \succ sp \right\}. \tag{3.7}$$

Every game must have at least one Pareto optimum. Often games have multiple optima, but there must always be at least one such optimum where all players adopt pure strategies.

In zero-sum games, all strategy profiles are strictly Pareto optimal.

From the point of view of system design Pareto optimality is an important concept. Clearly, outcomes which are not Pareto optimal should be avoided since they indicate that the results for some agents could still be improved without hurting any other agent.

3.4. Summary

Subsection 3.1 of this chapter has introduced four decision-theoretic approaches: Decision analysis, game theory, behavioral decision theory, and mechanism design. The capabilities, restrictions, assumptions, and focuses of each approach must be understood in order to choose the right method for a task. For the purpose of this work the main contribution of the four approaches is seen as follows:

- *Decision analysis* and *game theory* are two formal approaches to reasoning about decisions. These can be used for a model-based investigation of rational behavior of agents facing a given ATM-mechansim. (Important elements of the tool box of theories and methods have been more formally introduced in subsection 3.2 and can be used to develop a structured analysis of decision-making for ATM mechanisms.)
- *Mechanism design* provides formal concepts for the inverse question: 'How would a mechanism have to be designed to encourage a certain kind of behavior by self-interested ATM actors?'. The criterion of incentive-compatibility (see section 3.1.4) makes it possible to analyze if it is in the interest of an agent to report his information truthfully. This criterion will be transferred to ATM mechanisms and problems in this work.
- *Behavioral decision-making* research has pointed out certain limits to formal normative and prescriptive approaches. It has provided evidence of (and reasons for) significant deviations and biases in human decision-making. It has also shown that the mathematical approaches mentioned above must not be misinterpreted as perfect predictors of human behavior and that empirical research is equally necessary.

From the perspective of this work, despite some limitations, great potential lies in the application of the formal normative and prescriptive approaches of decision analysis and game theory for estimating and optimizing emergent system performance in ATM systems.

As [Keys07] argues, a balanced view to the field might be to acknowledge that while formal principles are not always optimal predictors of human behavior, they have an important role to play when reasoning about decision situations. Generally, a formal principle is a good approximation to substantive rationality if it can incorporate the most important factors involved in the situation [Keys07].

This section has introduced two system-optimality concepts with Hicks optimality and Pareto optimality. Further application-specific system-optimality criteria will be introduced and discussed in chapters 6 and 7.

Chapter 4 will address the question how complex and distributed ATM systems can be modeled to yield ~~to~~ tractable decision problems according to the requirements of the decision-theoretic and system-optimality concepts introduced in this chapter.

4. Net-Based Modeling and Analysis of Distributed Systems

This chapter introduces and discusses the background of this work in terms of modeling and analysis techniques. It provides the necessary foundations in Petri Nets (PN) and in particular in Coloured Petri Nets (CPN) [Jens97a], which build the modeling backbone for the design and analysis framework which will be developed in chapters 5 to 7.

The main contents of this chapter are:

- A discussion of the problem-specific requirements towards the modeling approach in the scope of this work (section 4.1),
- The identification of criteria for the selection of the Petri net modeling approach to drive the targeted framework (section 4.1),
- An introduction of the most important elements, characteristics, and definitions of Petri net formalisms (section 4.2), together with
- A simple example showing how PNs can be used for the modeling of games (section 4.2),
- The necessary background concerning the computer-based simulation, analysis, and visualization of CPNs (section 4.3).

Chapters 5 to 7 will recur to these contents to explain how decision theory and modeling support are integrated within the DAVIC analysis framework and how this serves the design of incentive-compatible ATM-systems.

4.1. Modeling Requirements

Requirement sources

Three main sources of modeling requirements seem to be particularly relevant with view to the targeted framework development. These are:

1. *Application domain characteristics* The typical features of the application domain as well as characteristics of the system of interest, which have to be represented and expressed in the model.

- Relevant domain characteristics have been discussed in chapters 1 and 2.

2. *Mapping to decision-theoretic representation* The need to derive a suitable formal representation from the system model for the purpose of decision-theoretic analysis.
   - The formalization of decision situations has been discussed in chapter 3.
3. *Execution of decision-theoretic analyses* The requirement to execute game-theoretic analysis and system-optimality analyses on the basis of the formalized decision problem.
   - Applicable solution concepts, algorithms and criteria for that purpose have been the topic of chapter 3.

The contribution of these three main requirement sources to the overall demands towards the modeling approach are depicted in Figure 4.1 on the layer 'Requirement Sources'.

Selection criteria for Coloured Petri Nets

In the context of this work Coloured Petri Nets [Jens07] have been chosen as the formalism and the related tool 'CPN Tools' [CPN 12] as the simulation and analysis tool to power the analysis framework. Petri nets (and in particular CPN) seem a suitable candidate modeling approach with view to the given requirements, particularly due to the following reasons:

1. *Support of domain characteristics*
   - Petri nets are a well-proven formalism for the modeling of distributed systems where concurrency, synchronization and resource-sharing play a major role [Jens97a].
   - PNs have been used successfully to model multi-agent systems (see e.g. [Mold97, Mold01, Kohl06, Kohl07]) and analyze communication protocols (e.g. [Liu07, Bill09]).
   - The structural properties of PNs and their typical application domain correspond well with the domain features and system characteristics of the ATM-problem as described in chapter 2.
2. *Support of mapping to decision-theoretic representation*
   - Petri nets allow the generation of state spaces and the software CPN Tools supports the automated generation of such state spaces for CPN models [Jens06, Kris00] (see also section 4.3 for state space methods).

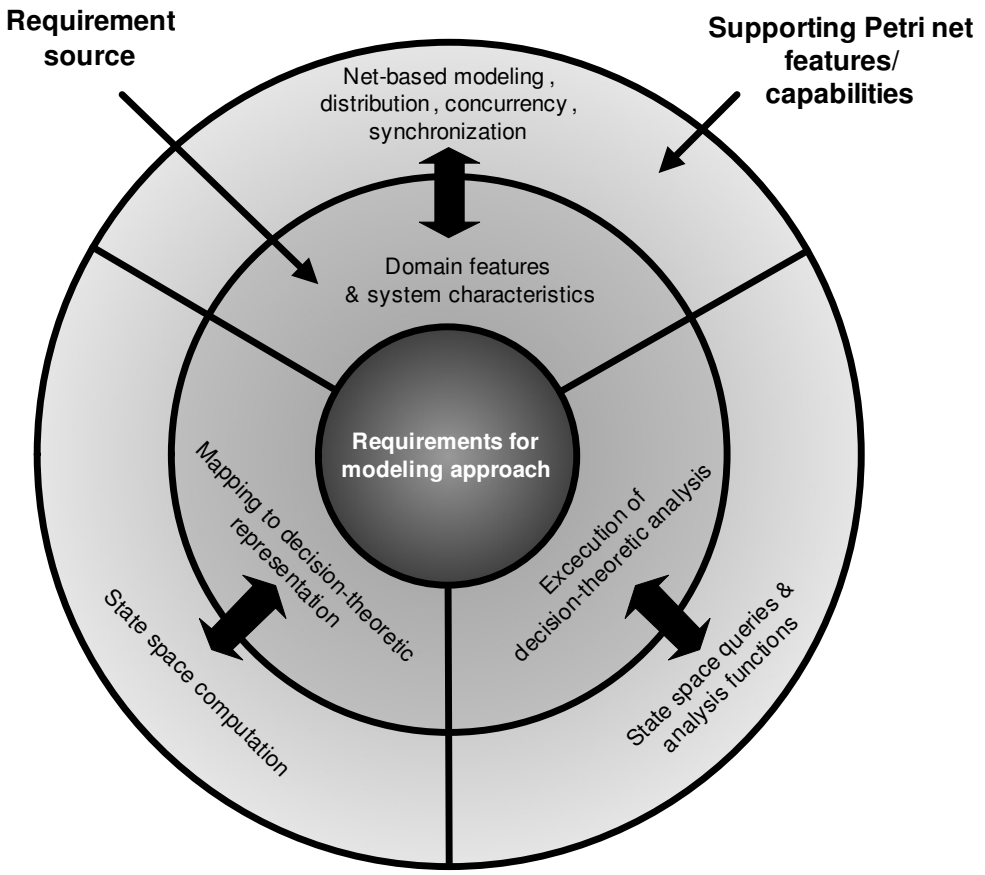

Figure 4.1.: Specific requirement sources for selection of modeling approach together with Petri net capabilities to support these requirements

- State space generation can be used to achieve a mapping from a highly compact representation of an ATM-mechanism as a CPN model to extensive state-space graph models, according to the requirements of decision trees and game trees, as discussed in chapter 3. Formerly [West06, Ober06] have discussed the potential of CPNs for constructing game trees from PNs and also [Tagi09, Clem06] for other classes of PNs.

3. *Support for decision-theoretic and system-optimality analyses*

   - CPN Tools has an integrated support for the execution of graph analysis algorithms on the computed state space.
   - Graph analysis features allow to construct and execute decision-theoretic analyses and system-optimality analyses as introduced in chapter 3, directly on the decision tree within the same tool [Jens06].

The outmost of the layers in Figure 4.1 summarizes how the three problem-specific requirement sources are addressed and supported by characteristic Petri net features and capabilities.

General CPN properties In addition to the features of PNs and CPNs here discussed in relation to the problem-specific modeling requirements of this thesis, Jensen [Jens97a] lists in a more general form some 'advantages' of CPN. It is acknowledged that due to the multitude of other formalisms the comparison remains implicit. Among other things important features are the graphical representation, the well-defined semantics, the general purpose language, the reliance on few but powerful primitives, and the explicit description of both states and actions.

Related work and applications of PNs and CPNs

Petri nets are a general-purpose modeling language and have been widely used in automation engineering (e.g. [Schn92]) also on applications of transport and traffic systems of all modalities. For the ATM/ATC domain, previously e.g. [Huck93] analyzed the potential of Petri nets for the observation of system states in airport traffic control and [Ruck97] employed Petri nets to model rule-based Pilot behavior. More closely related to this thesis work is research by Werther [Wert05, Wert06, Wert07], who used Coloured Petri nets to build integrated human-machine models of airports and air traffic controllers, with an interest in cognitive modeling. Subsequent research by Möhlenbrink [Mohl07, Mohl10, Mohl11] focused on the modeling and analysis of heuristics for decision-making of air traffic controllers (also for the airport domain) and applied CPN to examine human decision-making in a microworld approach. Primarily with an interest in technical cognition and also learning for robotics, [Gamr09b, Gamr11] recently employed Higher Level Petri Nets (Reference Nets) to implement a system-theoretic modeling approach developed in [Soff03]. A CPN-based microworld approach to support the empirical behavioral analysis of human decision-making was also employed in [Hass09, Ober10] to analyze human conflict-solving in approach and en-route air traffic control. Also [Kova05] and [Huan10] used Coloured Petri Nets to model aspects of the ATM system.

4.2. Petri Net Formalism and Definition

Petri nets in their original form (as Condition/Events Nets) go back to the foundational work of Carl Adam Petri in [Petr62]. Based on this original formalism, a wealth of extended formalisms has been developed.

For this section, the following approach is taken to introduce the necessary Petri net definitions and background:

1. In subsection 4.2.1 PT-nets (Place/Transition Nets) are presented as an example of a basic formalism, close to the original definition in [Petr62].
2. In subsection 4.2.2 CP-nets (Coloured Petri Nets) [Jens07] are introduced as an extended high-level formalism. CP-Nets are also the modeling approach which is employed in the sequel of this work.
3. In subsection 4.2.3, a brief outlook on selected high-level formalism extensions to PN is given.

More detailed bibliographical remarks including the historical development stream from Condition Event (CE-nets) covering Place Transition Nets (PT-nets), Predicate Transition Nets (PrT-nets) to CP-nets can be found e.g. in [Jens97a]. A comprehensive overview of available Petri net tools and summary of their capabilities can be found in the online database [Info12].

Example game

Where possible throughout the sections 4.2.1 to 4.3, the presented theoretical concepts and their applicability for the modeling of games are illustrated on the following example of the 'Game of Chicken'. All explanations related to this example are framed in boxes to distinguish between definitions, terminology, and application example.

Demonstration example: 'Game of Chicken' (GoC)

The chicken game is a model of conflict for two players which has been subject to intensive research in game theory (see e.g. [Born97]). An interpretation of the game is that the two drivers drive towards each other on a collision course, performing a test of courage. One driver must swerve, otherwise both will end up in a crash. If one driver swerves and the other does not, the one who swerved will be called the 'chicken' (German: 'Feigling'). The other driver will be the winner. The corresponding payoff matrix of the normal-form game may be set up as follows:

		B	
		swerve	*straight*
A	*swerve*	0,0	-1,+1
	straight	+1,-1	-10,-10

The game-theoretic analysis of the situation yields two pure-strategy equilibria (swerve/straight and straight/swerve) plus a symmetric mixed-strategy equilibrium, which is 90% swerve, 10% straight for the given payoffs.

4.2.1. Place/Transition Nets

Petri nets consist of a static and a dynamic component. For Place/Transition Nets (PT-nets) both components together form the five-tuple $PT = (P, T, I, O, M)$.

Static component

The static component of PT-nets consists of three basic elements: Places, transitions, and arcs, forming a particular kind of bipartite graph.

The standard interpretation of these entities is that places represent conditions and transitions represent events. Arcs represent the causal relationships between conditions and events.

Regarding the types of arcs, input arcs and output arcs are distinguished. Input arcs point from places to a transitions, while output arcs point from a transitions to places.

Places, transitions, and arcs are termed the static component, as their topology is not affected by state changes of the net, e.g. during simulation.

Formally, the static net-component can be written as a quadruple (P, T, I, O), where [Zhou09]

1. the set $P = \{p_1, p_2, \ldots, p_m\}$ is a finite set of m places,
2. the set $T = \{t_1, t_2, \ldots, t_n\}$ is a finite set of n transitions, $P \cup T \neq \emptyset, P \cap T = \emptyset$,
3. the function $I : P \times T \to N$ is an input function that defines the directed arcs from places to transitions, where $N = \{0, 1, 2, \ldots\}$,
4. the function $O : P \times T \to N$ is an output function that defines the directed arcs from transitions to places.

Static component of GoC

In Figure 4.2 the static PT-net component of the GoC game is shown in graphical representation. Places of PN are typically drawn as circles while transitions are drawn as bars or rectangular boxes. Directed input and output arcs connect these elements. In the example, the transitions represent the players' decisions and the distribution of payoffs (modeled as two sequential steps). The places symbolize different possible states of the game.

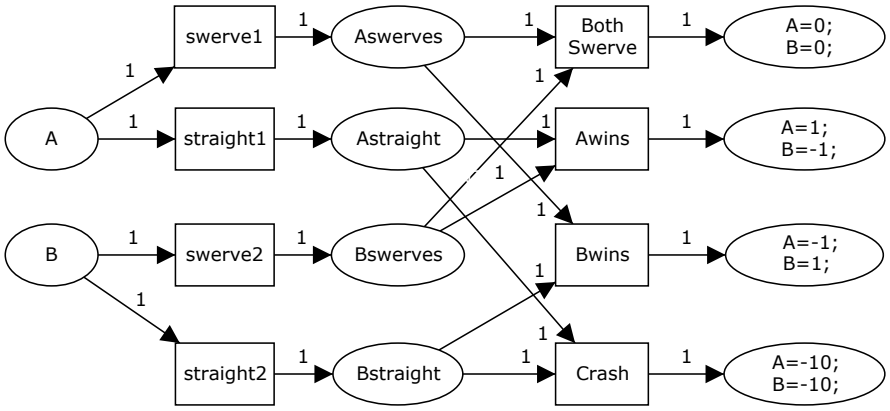

Figure 4.2.: Static component of 'Game of Chicken' represented as PT-net

Dynamic component

The dynamic component of a PT is its *marking*. Formally, the marking $M : P \to N$ represents the number of tokens in places of P. It assigns to each place a non-negative integer. M_o denotes the initial marking of the net.

The distribution of tokens (and thus the dynamic state of the net) is changed by the firing of transitions. The state changes can alter both the position and number of tokens in the net. The changes are interpreted as discrete state transitions.

Dynamic component of GoC

Figure 4.3 represents the same (formerly unmarked) static net structure as Figure 4.2, now including the dynamic component. Three different markings are shown, which are represented by black dots (tokens) in place p, as typical of PT-nets,. The state transition between the two markings corresponding to subfigures State 1) and State 2) is caused by the firing of transitions 'straight1' and 'straight2'. The state transition between State 2) and State 3) is caused by the firing of transition 'Crash'.

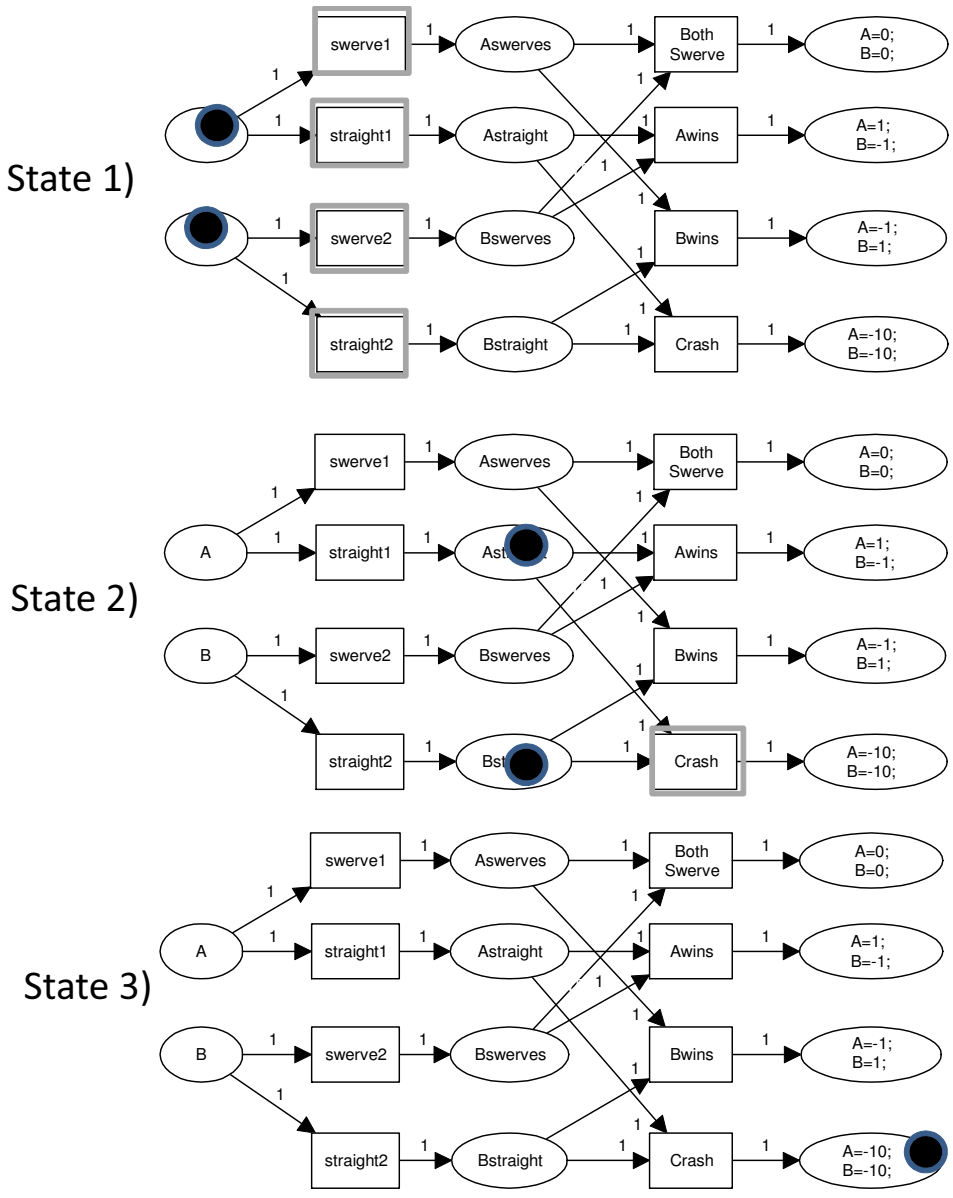

Figure 4.3.: Dynamic component of the 'Game of Chicken' for the PT-net

Enabling of transitions and firing rule

The subsequent states in which a PN can jump from a given state are defined by the topology of the net and the firing rule. The firing rule defines the conditions for the transfer of tokens along the directed arcs. A transition t can only be fired if each of the input places of this transition contains at least as many tokens as required by the input function $I(p,t)$ for the corresponding input arc. If this is the case, the transition is said to be *fireable* or *enabled*.

The firing of a transition means the withdrawal of the number of $I(p,t)$ tokens from each of the input places of the transition and the addition of the number of tokens $O(p,t)$ to each of the output places of the transition.

As simple PNs are discrete event systems, the firing of transitions is indivisible and is considered to have zero duration.

Concurrency and non-determinism

The notions of concurrency and non-determinism in Petri net systems are closely related to the concept of enabledness.

- *Concurrency:* One of the specialties of Petri nets is the modeling of concurrent(i.e. parallel) processes and synchronization. Several transitions $T_{n...m}$ may be concurrently enabled for a given marking. These transitions may occur 'at the same time' or 'in parallel'. On the other hand it is also possible that two enabled transitions have bindings with joint sets of tokens. In this case the transitions are said to be 'in conflict'. Only some of the transitions can fire, as their firing will remove tokens which would also be needed by the other ones.
- *Non-determinism:* An important property of PN formalisms is that they can model non-deterministic behavior. There is, at least for simple PN, no priority rule for the decision of the order of firing between several enabled transitions. The selection is considered to be random. Since firing of transitions (and execution of tasks) is non-deterministic and tokens can be located anywhere over the distributed net model, Petri nets are a widely used tool for the modeling of concurrent behavior in distributed, decentralized systems.

Enabling, firing, concurrency and non-determinism for GoC

In Figure 4.3 State 1), four transitions are initially enabled. However, the upper two and lower two transitions are in pairwise conflict. Firing one of them will remove the necessary token for the other. Within the pairs, the firing is non-deterministic as the outcome of the choice 'swerve' or 'straight' for each of the players is open. Between the pairs, firing is concurrent (in the same time step), players take their decisions in parallel. The second column of transitions will force a synchronization of the process, since both players' decisions are needed to determine the final outcome of the game.

4.2.2. Coloured Petri Nets

Coloured Petri nets (CP-nets) [Jens97a] are an extension of PT-nets. They present a class of Higher Petri Nets (HPN). Their main goal is to achieve a more compact representation of the modeled system. This compactness of CP-nets is realized primarily by equipping each token with an attached data value, the token colour. The tokens (and places) also have defined data types, the coloursets. The colourset can be arbitrarily complex, analogous to the datatypes in modern programming languages. They can embody e.g. lists or nested data structures.

The attribution of tokens with data values (as opposed to using 'plain black' tokens in PT-nets, see Figure 4.3) makes it possible to handle additional data in the net without making the net structure more complex. The data values are processed and changed by net expressions formulated inscription language of the net.

Formally , a non-hierarchical CP-net is a tuple $CPN = (\sum, P, T, A, N, C, G, E, I)$ which satisfies the following requirements (see [Jens97a]):

1. The set $\sum$ is a finite set of non-empty types, called **colour sets**.
2. The set P is a finite set of **places**.
3. The set T is a finite set of **transitions**.
4. The set A is a finite set of **arcs** such that: $P \cap T = P \cap A = T \cap A = \emptyset$.
5. The function N is a **node** function. It is defined from A into $P \times T \cup T \times P$.
6. The function C is a **colour** function. It is defined from P into $\sum$.
7. The function G is a **guard** function. It is defined from T into expressions such that:
   $\forall t \in T : [Type(G(t)) = \mathbb{B} \wedge Type(Var(G(t))) \subseteq \sum]$.
8. The function E is an **arc expression** function. It is defined from A into expressions such that:
   $\forall a \in A : [Type(E(a)) = C(p(a))_{MS} \wedge Type(Var(E(a))) \subseteq \sum]$ where $p(a)$ is the place of $N(a)$.
9. The function I is an **initialization** function. It is defined from P into closed expressions such that $\forall p \in P : [Type(I(p)) = C(p)_{MS}]$.

CP-net elements, properties and terminology The following properties, elements, and terms related to CP-nets are highlighted:

- *Net inscriptions:* In addition to its graphical components, the CP-net is defined by net inscriptions. These net inscriptions allow to define the colour sets and initialization of places, to formulate arc expressions (with variables) for input- and output arcs, and to define the guard functions.

- *Inscription language:* The formal CP-net definition (see above) does not assume any particular language. For the CPN models and the simulator used in this work, the inscription language is the functional language CPN ML based on Standard ML [Harp05].
- *Code segments:* With the firing of transitions, code segments can be executed e.g. to perform mathematical calculations, data processing, or realize input/output operations (e.g. read from file, write to file).
- *Guard function:* A transition can be assigned a so-called guard function, as a Boolean expression. Besides a sufficient marking of the input places of the transition, this guard function has to evaluate to true for a given binding for the transition to be able to fire.
- *Hierarchy/Modularization:* CP-nets support the use of hierarchical structures and modularization, allowing the modeler to combine several smaller CP-nets into a larger net. The basic means of achieving hierarchy is the 'substitution' of transitions. The second means of realizing modularization are 'fusion places'. These specify a set of places which are considered to be identical (i.e. will always contain the same set of tokens).

With the support of complex data types, hierarchy, modularization and textual code segments, CP-nets allow the modeling of systems which would be practically intractable as PT-nets due to their size and complexity, although their representation as PT-nets is in principle possible.

CP-net elements and extensions for GoC

In Figure 4.4 once again the GoC is implemented, this time however in the language of CPN. In comparison to the previous PT-net implementation, the following features of CP-nets can be observed:

- Players, available choices, decisions, and payoffs are represented as distinct data types (see labels close to places)
- Both players A and B share a single transition 'decide' for the decision process, a single firing of this transition implements both players' decisions.
- The 'act' transition is a substitution transition implemented on a subpage, the model is hierarchical (the Figure shows superpage and subpage).
- Several features such as the outcome of a decision or the value of the payoffs are transported in the coloured tokens. This reduces the number of necessary places, e.g. there is now just one terminal place 'payoff'.

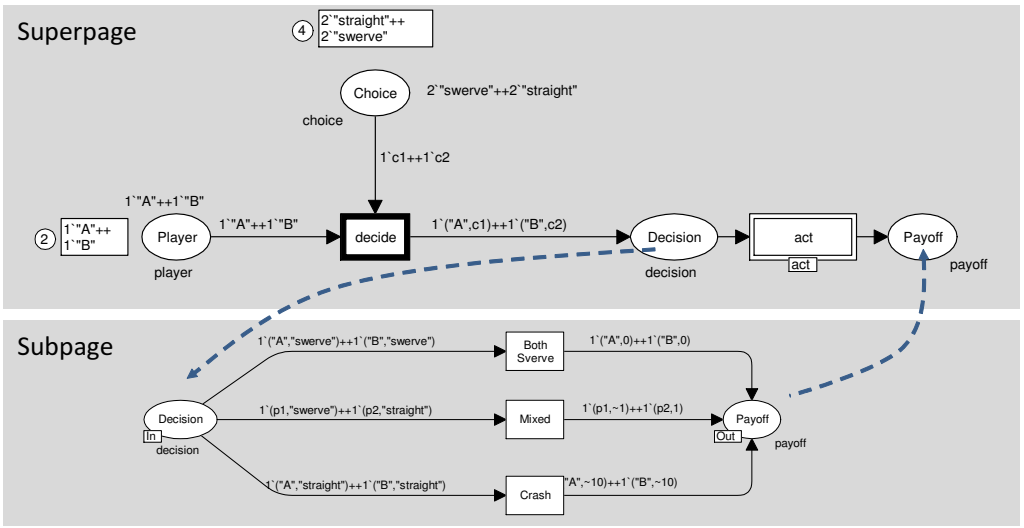

Figure 4.4.: 'Game of Chicken' as a hierarchical Coloured Petri Net

Note that for the demonstrated simple GoC game with two players, the reduction achieved by moving from PT-nets to CP-nets is not big. For more complex applications (e.g. with more players) reductions tend to be much more significant.

4.2.3. Petri Net Extensions

In addition to the introduction of Coloured Petri Nets, a great number of other Petri-net based modeling formalisms has been discussed in literature as extensions to the original formalism (see e.g. [Davi05]). Some important extensions are:

- *Time-based nets:* The formalism is extended with a support for the modeling of timed processes and delays caused by the execution of certain actions. In CP-nets, timed nets are implemented by the attribution of transitions or output arcs with time durations.
- *Stochastic Petri nets:* Stochastic Petri nets are a subsidiary of timed nets. Stochastic nets (see. e.g. [Haas10]) introduce adjustable random variables into the formalism in order to represent the duration of activities, or the delay before events. The random delays affect the probabilities of the next state transition of the net.
- *Priority nets:* Priority nets add priorities to transitions, so that one transition cannot fire if a higher-priority transition can fire. Priority nets are used when it is desirable to choose between a number of enabled transitions. Then the priority rule eliminates the non-deterministic, random firing-rule of the regular nets.

- *Continuous and hybrid nets:* Alternatively to the purely discrete PN formalisms described above, there are variants that support modeling of continuous and hybrid (discrete-continuous) processes [Davi05]. The basic feature of a continuous net is that the marking of a place is a real number instead of an integer. The firing of transitions takes place in a continuous flow. Hybrid nets are combinations of continuous and discrete Petri nets.

Several authors (e.g. [Huck93]) have pointed to the trade-off between the expressive power of a modeling formalism and its potential for analysis. That is, the more complex the extensions to the PN formalism, the more difficult the formal analysis and proof of specific net properties (see also following subsection) tends to become. While expressive power is gained, some extensions thus sacrifice a strong point of PNs in their original form.

4.3. Petri Net Simulation and Analysis

This section points to the difference between simulation and analysis as 'execution modes' for PNs. It considers methods for the generation of state spaces (SS) and the analysis of SS with view to specific net properties. Some of the consideration can be generalized over a range of (or all) types of Petri nets. Others apply primarily to the Coloured Petri Net formalism and the computer-based tool CPN-Tools used in this work.

4.3.1. Simulation vs. Analysis

Simulation The term simulation is used in this work to describe the execution of a single trajectory of sequential steps with a PN. Each simulation step is defined by a single binding element. The resulting occurrence sequence is one out of all possible sequences starting from a given marking M_0.

Simulation Modes: With available software tools (notably CPN Tools), simulation can be executed in various modes. These include:

- *Manual mode,* where the modeler selects each binding manually and observes the effect.
- *Automatic mode,* where a high speed random selection from a set of enabled transitions is performed in each step and the output e.g. written to a file.
- *Batch mode,* where a very high number of simulation runs from different initial markings M_0 is automatically executed.

Use and limitations: Simulation helps to increase the understanding of the modeled system. Especially batch simulation can also provide valuable data regarding the expected 'average' system performance. Simulation alone however also has certain

conceptual limitations for the evaluation of non-deterministic systems. Sequential simulation, for example, cannot render

- a complete proof of dynamic properties of the model,
- absolute minima or maxima of process time,
- the proof that the system is free of potential deadlocks.

For that purpose, formal analysis methods are needed.

Formal analysis The notion of (formal) analysis is used in this work to describe activities aiming at

- the proof of dynamic poperties of a system or
- statements about the entire set of system states and about all trajectories the model may reach or perform.

The suitability of PNs for formal analysis is one of their major strengths. Two main forms of formal analysis are commonly used for PNs:

- *Invariants* are a form of analysis based on methods from linear algebra. It is distinguished between place invariants (the formulation of expressions which are satisfied for all markings) and transitions invariants (the expression of occurence sequences that have no total effect) [Jens97b].
- *State space methods* are a form of analysis based on methods from graph theory [Kris00, Jens06].

This work exclusively uses state space methods, which are more easily automated and better supported by tools.

The idea of state space methods is to calculate a graph which contains one node for each marking (state) that can be reached, and one arc for each occurring binding element. On a complete state space graph, properties of interest can be checked/inspected by systematically traversing the graph.

4.3.2. State Space Definition and Generation

State space definition

The state space SS of a Coloured Petri Net is a directed graph $SS = (N_{SS}, A_{SS})$ with arc labels from the set of binding elements BE, where [Jens09]:

- $N_{SS} = \mathcal{R}(M_0)$is the set of nodes reachable from the initial marking.
- $A_{SS} = \{(M, (t, b), M') \in N_{SS} \times BE \times N_{SS} |\ M \xrightarrow{(t,b)} M'\}$ is the set of arcs.

SS is finite iff N_{SS} and A_{SS} are finite.

The state space graph SS is alternatively named occurrence graph or reachability graph.

As the definition points out, the state space is a function of the initial marking M_o. In general, a different state space (and different state space properties) will result from a state space generation if the same static PN structure but a different initial marking $M_{o*} \neq M_o$ is used.

State space generation

The following information regarding state space generation is relevant in the context of this work:

- *Search algorithms:* The generation or exploration of state spaces can be conducted using standard algorithms for traversing graphs, e.g. breadth-first or depth-first traversal methods are possible. Conducted in this way, the complexity of the search algorithms itself is low and algorithms are well documented.
- *State space explosion:* An important challenge with state space methods tends to be the problem of state space explosion. The state space of interest may have more states (and therefore need more memory) than can practically be handled by the executing analysis software and/or computer system. To cope with this problem, refined state-space analysis-methods and state-space reduction-techniques have been developed. Examples are the sweep line method [Gall02] and ComBack method [West07].
- *Avoiding dispensable states:* To facilitate state space analysis, PN model construction should avoid unnecessary intermediate states as far as possible. This can be done e.g. by grouping a number of sequential steps together in one code segment.
- *Partial state spaces:* Instead of calculating a singular full state space, it may be possible to substitute the problem by multiple partial state spaces. Section 4.3.4 presents a state space tool extension to CPN Tools which technically enables the handling of multiple partial state spaces.

State space visualization

In addition to the execution of formal or quantitative analysis, the graphical representation of state spaces supports the understanding of system behavior. State space visualization is primarily helpful for systems with a small number of states. For large systems often only partial state space visualizations can be practically handled.

For the construction of graphical representations of state spaces, computer-based visualization tools are available. For CPN Tools e.g. an interface to the GraphViz

library [ATT 12] exists. GraphViz contains both templates for automatic drawing of standard graph types and ample opportunities for creating customized graph types.

State space for GoC

Figure 4.5 shows a state space graph corresponding to the CPN version of the GoC previously introduced in Figure 4.4. The state space consists of a tree structure with nine nodes (the initial node, four intermediate nodes, four terminal nodes). The four terminal nodes represent the four outcomes of the game. The eight arcs are labeled with the corresponding bindings of the firing transitions. Within the entire state space, one particular simulation trajectory is highlighted by bold arcs. This trajectory represents the specific occurrence sequence from the individual simulation experiment given in Figure 4.3, State 1) State 2) and State 3).

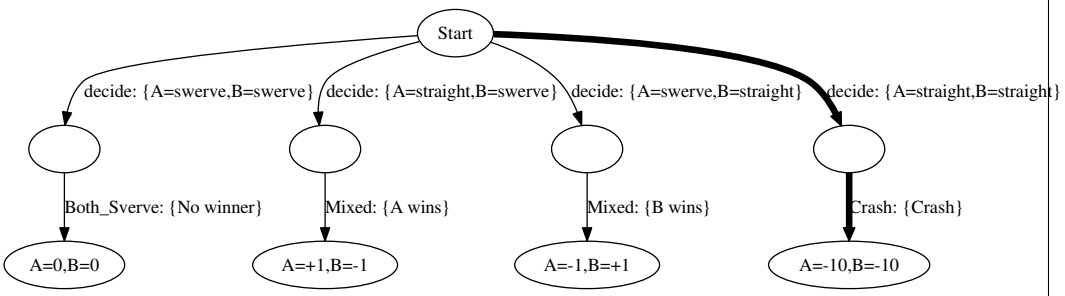

Figure 4.5.: State space for game of chicken

With the automatic generation of this state space, Petri nets support the step from the specification of the rules of the game to the construction of the full game tree which is ready for the formal decision-theoretic analysis as described in chapter 3. Note that the game tree is in this example case very limited in size. It could still be constructed manually without employing a CPN Model. However, the state spaces for relevant collaborative ATM applications can become very large. This is true even for a moderate rule base which can be expressed by a small CPN Model. Then, the direct specification of the game tree can quickly become unfeasible, and the advantages of specifying the interaction in the form of a CPN pay out.

4.3.3. State Space Analysis and Net Properties

State space analysis makes it possible to verify, i.e. prove that the model (or certain places or transitions of the model) possess a certain formally specified property (for the given initial marking).

Net properties State space analysis for Petri nets is frequently used to verify e.g. the following properties [Davi05]:

- *Reachability:* Reachability determines if it is possible to get from one node in the directed state space graph to another node.
- *Boundedness:* A place P of a Petri net is called *k-bounded* for an initial marking M_o if there is a natural integer k such that for all reachable markings from M_o the number of tokens is not greater than k. A PN is bounded for an initial marking M_o if all the places are bounded for M_o, it is $k - bounded$ if all individual places are $k - bounded$.
- *Liveness:* A transition T of a Petri net is said to be live, if it is potentially fireable in each reachable marking M. This means that in each marking M, there exists a sequence of steps including the firing of transition T.
- *Deadlock:* A deadlock (or sink state) is a marking in which no transition can be fired. A PN is said to be deadlock-free for an initial marking M_0 if no reachable marking $M(M_0)$ is a deadlock (or dead marking).

If a proof of a net property does not hold, the state space search can produce the counterexample. This makes it possible to inspect the specific model state(s) for which the proof fails.

Realization of state space analyses Regarding the practical implementation of analysis algorithms the following aspects are to be considered:

- *Standard queries* For standard graph analyses, such as the ones defined above, ready-built analysis functions can be found in state space analysis libraries, which provide results without any programming. For CPN Tools, these are defined in [Jens06].
- *Custom queries* For more specific needs, the construction and programming of self-made, customized analysis routines is needed. This is the case for the more complex decision-theoretic analysis routines employed in this work. In CPN Tools, it is possible to program arbitrary custom queries in CPN ML on top of the functions of the standard library. Doing this, one can exploit the available generic graph analysis functions for graph traversal, filtering of states, and all sorts of low-level routines.
- *Computational performance* Computational performance can become a major practical issue and limitation to state space analysis, in the same way as for the problem of generating state spaces in the first place. The severity of the problem depends on the size and complexity of the state space and the computational order of the analysis algorithm.

4.3.4. Closed-Loop State Space Analysis and Simulation

Limitations of CPN Tools' built-in functionality

The software CPN Tools which is employed in this work supports the execution of CPN models both in simulation and in analysis with a wide set of functionalities. However, the following constraints exist regarding the built-in functionality:

- CPN Tools treats simulation and analysis as two mutually exclusive options. These are employed one at a time but not in direct interaction or closed loop (This is in fact the common approach for Petri net research in general).
- CPN Tools has some built-in support for batch simulations. There is no support for batch mode state space analysis, however. State space analysis for each specific marking has to be started manually via the graphical user interface.

Closed loop simulation and state space analysis extension

In order to remove some of these limitations in CPN Tools, a novel method was presented in previous work by the author of this thesis [Ober08b, Ober08a, Gamr09a]. A software extension was developed for the integrated application of simulation and state space analysis within a single automated run of a CPN model in CPN Tools. The approach allows the analyst

1. to automatically and repetitively execute simulation, state space generation and analysis,
2. to use the results of each alternating simulation or state space analysis phase for the initialization of the following phase,
3. to form a closed loop between state space analysis and simulation,
4. to split one full state space into multiple partial state spaces.

The Closed Loop Simulation and State Space Analysis approach (ClSimSSA) supports the automatic handling of simulation and state space analysis in a more flexible manner and leads to new possibilities in evaluation approaches for PNs.

The presented CPN extension has been applied to a number of different domains, such as an example of a cognitive technical system in [Ober08a], an evaluation of human conflict resolution in air traffic control [Ober10, Hass09] and the analysis of arrival management in [Ober09]. The latter presents the technical basis and predecessor for the decision-theoretic and game-theoretic analysis in chapters 6 and 7 of this work.

5. Framework for Design And Verification of Incentive-Compatible ATM Systems (DAVIC)

This section presents the framework for the Design And Verification of Incentive-Compatible ATM-Systems (DAVIC framework) which has been developed in this work.

The DAVIC framework defines a novel structure for an integrated modeling and analysis approach. This approach is based on decision-theoretic- and system-theoretic analysis criteria and a net-based modeling of distributed ATM-systems. The framework is usable for a wide range of distributed systems in which emergent system behavior is determined by agents' interests and choices. It will be applied to the concrete example of a planning mechanism for cooperative arrival management in chapter 6. Chapter 7 will present the results of the analysis of this application.

The DAVIC framework is proposed as a potential tool to fill the methodological validation & verification gap outlined in chapter 2. It differs from existing approaches and work on incentives and interest in ATM systems (section 2.4 and chapter 3) particularly through its seamless integration of modeling support with decision-theoretic analysis tools and system-optimality criteria. This allows a tight coupling with state-of-the-art system engineering and requirement engineering methods.

The framework itself is considered to be a verification tool rather than a validation tool, since as it verifies the compliance of a system with a set of specific, formally defined criteria coming from decision theory and system-theory. While this can prepare and support validation activities, the scope of validation activities in ATM generally includes a wider set of user- and operational concerns and usually encompasses empirical studies with human operators in the loop.

The presentation of the framework in this section starts with a discussion of the rationale for its development and its underlying requirements and assumptions in section 5.1. This is followed by an overview of the framework structure and a detailed explication of its individual components and their interrelations in section 5.2.

The section is concluded by a discussion of how the DAVIC framework can be used to achieve the optimization of incentive-compatibility and performance in distributed ATM-systems (section 5.3). This goal is achieved by conducting a systematic search over the design space of defined classes of mechanisms and operational scenarios.

5.1. Rationale for DAVIC Framework Development

The purpose of the developed DAVIC framework is to provide a structure for the analysis and verification of incentive-compatibility in distributed ATM-systems. To do so, an integrated approach of decision-theoretic techniques and net-based modeling is developed. The framework builds upon - and is motivated by - the problem statement as established in sections 1.2 and lack of validation & verification instruments and tools as described in section 2.4. As a potential contribution to closing the method gap, the framework integrates decision-theoretic techniques and system-optimality concepts (as discussed in chapter 3) with Petri-net-based modeling techniques (as discussed in section 4). These are fused to form a coherent and reusable structure and analysis approach.

In generic terms, the purpose of a framework is to provide "a supporting structure around which something can be built" [Fram12]. A more system- and software-oriented definition by Fayad [John88] describes a framework as "a reusable design of a system that describes how the system is decomposed into a set of interacting objects [...]. The framework describes both the component objects and how these objects interact. It describes the interface of each object, and the control flow between them. It describes how the system's responsibilities are mapped onto its objects".

In line with these definitions, the DAVIC framework proposes a number of fundamental components which support the analysis and optimization of incentive-compatibility. Further, it describes the distribution of responsibilities (sub-tasks of the verification and optimization procedure) of these components and the interaction of the components via defined interfaces.

The framework is designed to satisfy a number of key requirements and objectives. In the development, the following rationales thus play a decisive role:

- *Modeling and analysis integration:* The framework shall provide a closely integrated structure for modeling, analysis, and optimization of incentive-compatibility in distributed systems, rather than a collection of isolated tools for the different tasks. Support for modeling complex mechanisms and analyzing decisions within this mechanisms stand side by side, are of equal importance, and are closely related.
- *Confrontation of agent view & system view:* The framework shall provide structured support for the contrastation of diverse decision-theoretic- and system-optimality criteria. It shall be easily extendable with regard to the definition of additional analysis criteria for future applications.
- *Engineering approach:* The framework shall support an engineering-oriented approach and design process. Reusable and standardized simulation and model checking procedures are to be favored over specialized analytical proofs which are applicable only to highly specific classes of mechanisms.

- *Generic structure:* The generic framework structure shall be independent as far as possible from the specific properties of the mechanism to be analyzed. Thus while the selection of specific criteria for agent rationality and system optimality will naturally depend on the application domain and context, this shall not alter the basic responsibilities and interactions of the framework components.
- *Modularity:* The framework shall be modular and support re-usability of all its components wherever possible. This concerns the reuse of mechanism models for analysis purposes independently of decision-theoretic problems and the application of the same decision-theoretic and system-theoretic analysis criteria for a wide range different mechanisms, independently of a particular application domain.

5.2. Framework and Components

This section presents the resulting modeling and analysis framework DAVIC (Design and Verification of Incentive-Compatible ATM-Systems) which has been designed according to the above rationale. The function and properties of the individual framework components which satisfy the presented requirements are discussed below.

An overview diagram of the framework structure and its components is provided in Figure 5.1. The Figure shows that the framework consists of eight different entities representing software components and definition layers. These entities are characterized briefly in the following. They are then discussed in more detail in the subsequent sections.

1. *Application Layer:* Defines in informal, natural language the application context, potential mechanism structures, and expected operational scenarios, as well as the requirements for operating the system successfully.
2. *Model Layer:* Formalizes the potential mechanisms and operational scenarios into executable model structures and initial model conditions.
3. *State Space:* Represents a directed graph containing all reachable system states for a given system model and simulation scenario.
4. *State Space Interface:* Makes the state space accessible through generic formalizations of games and of decision problems.
5. *Agent-Rationality Component:* Implements decision- and game-theoretic solution concepts in order to identify rational outcomes of a game.
6. *System-Optimality Component:* Implements designers' criteria of optimal system performance by analyzing the state space.
7. *Analysis Layer:* Integrates agent- and system view by performing set operations on model states. Checks formal conditions (proofs) and evaluates quantitative analysis criteria.

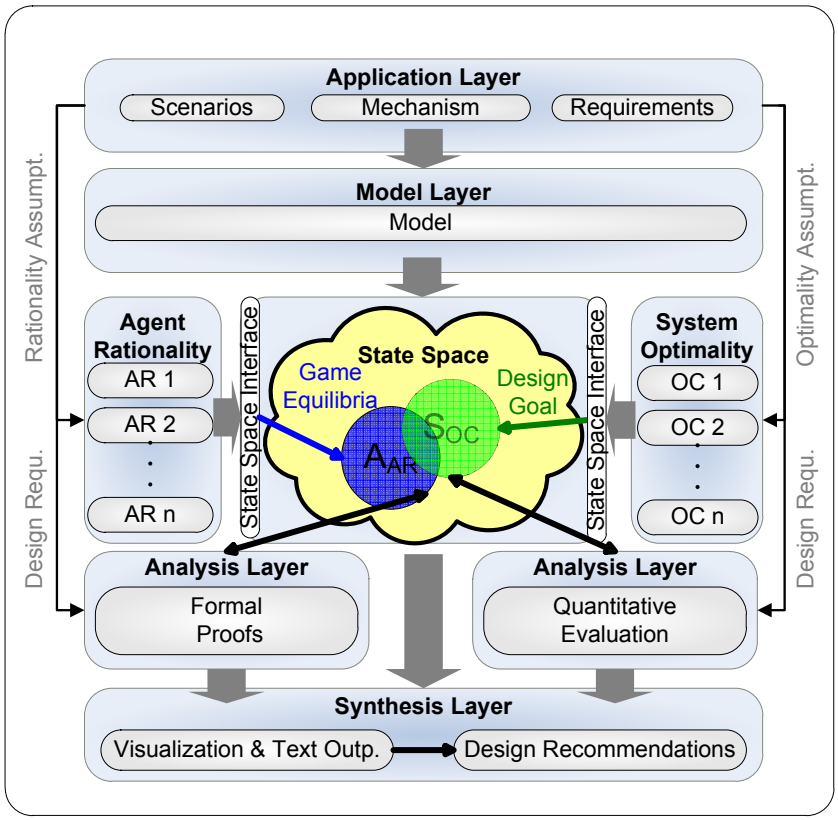

Figure 5.1.: Framework for Design and Verification of Incentive-Compatible ATM Systems (DAVIC)

8. *Synthesis Layer:* Visualizes and synthesizes analysis results and derives conclusions in terms of design recommendations.

5.2.1. Application Layer

The starting point for each analysis and optimization project with the DAVIC framework is the collection of a comprehensive set of information on the system's application context and its planned concept of use. In terms of the Structured Planning Framework of the E-OCVM Standard (section 2.3), this important task corresponds closely to activities listed under E-OCVM Step 0, 'State Concept and Assumptions' [Euro04]. The aim of this step is to acquire a thorough understanding of

1. the system's purpose, statement of need and the identified ATM problem which the system is supposed to solve (the problem domain), and
2. the proposed solutions to the problem, i.e. a description of the operational concept and the role of the assumed technical planning system(s) within the concept (the solution domain).

Mechanism, scenarios, and requirements

For the specific goal of validating incentive-compatibility and performance effects of a developed concept, the information on potential mechanism structures, expected operational scenarios, and requirements for successful system operation are of particular importance. The documentation thus should with sufficient depth treat the following issues:

- The mechanism, including
  - the concept of the control- and planning system which allocates resources within the system, and
  - the rules for interacting with the planning system, such as a generic formulation of the available use cases and choices.
- The scenarios, including
  - operational situations and initial conditions which the system is thought to encounter,
  - choices available to the actors within the scenarios, and
  - a description of how the system is supposed to react within the scenarios.
- The requirements, including
  - the designer's expectations with regard to individual actor behavior, and
  - requirements with regard to emergent system performance.

Type of output

The type of output produced by defining the application layer is generally a textual description in natural language. This is in contrast to the other DAVIC components, which mainly consist of software modules to execute specific functions within the framework. On the application layer, the description does not have to be highly formalized and does not have to contain technical or mathematical details of the system specification. Rather, these descriptions should be easy to read and understand as this quality determines the possibility of validating the identified user requirements and system purpose with stakeholders and subject matter experts.

Dependencies

In the following processes involved in applying the DAVIC framework, the initial mechanism description developed on the application layer is a crucial input for the development of the formal process model.

The scenarios will determine the initial model conditions (i.e. operational situations) which are considered in the analysis.

Further, the identified performance requirements will guide the selection and development of system optimality criteria.

From an analysis point of view, the application context and system surrounding determine the selection of appropriate rationality concepts (decision-theoretic or game-theoretic solution concepts) e.g. according to assumptions on the availability of information to actors in the specific situations.

The dependencies are indicated graphically in Figure 5.1 by arrows and links between the respective components. The nature and consequences of these dependencies are focused on in more detail below in the respective sections.

5.2.2. Model Layer

Building upon information gathered on the application layer, the model layer formalizes the system structure (planning mechanism and actor behavior) as an executable CPN model, the *system model* (SM). It also formalizes the scenarios defined on the application layer by a mathematical description of input data that serves to initialize the CPN model structure with an initial marking, a *scenario specification* (Sc).

Due to the intrinsic nature of any modeling activity, the model necessarily and deliberately presents an abstraction from reality and reduces the richness of reality and the description provided on the application layer. This abstraction concerns both the model structure itself (the system model) and the type of information used to initialize the model for specific system scenarios (scenario specification).

The engineer conducting the modeling- and verification task has the responsibility to judge which characteristics of the system and scenarios are likely to influence the choices of actors in reality and what will therefore be of interest for an analysis of decision-theoretic criteria and incentive-compatibility. These elements are to be represented as part of the model. All other elements, which are not expected to impact incentives for agents, should be omitted, in order to keep the model as concise and make subsequent computational analysis as efficient as possible. During this selection process, assumptions regarding the relevance and representativeness of the model and its parts should be made transparent and documented.

System Model (SM)

The structure of the CPN *system model* within the DAVIC framework generally consists of three distinct parts

1. *Mechanism model (MM):* A model of the (automated) planning system and resource allocation mechanism (the mechanism center)
2. *Agent behavior model (AM):* A model of agent behavior and available agent strategies (distributed actors)

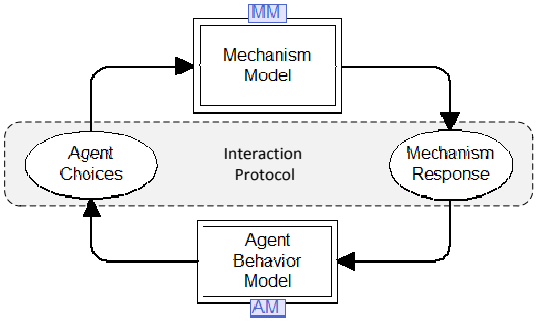

Figure 5.2.: High-level system model (SM) in CPN, representing the interaction between mechanism model (MM) and agent-behavior model (AM)

3. *Interaction protocol (IP):* The protocol used by agents to interact with the planning system and the feedback loop from the planning system to communicate the system response

Figure 5.2 shows a graphical representation of a CPN model with these system parts.

The two transitions represented by rectangles encapsulate the behavior of the planning system (mechanism center) and the agents (agent behavior) respectively. Both transitions are substitution transitions (indicated by the labels attached to the transition), implying that the detailed model behavior is specified by a model subpage linked to this transition.

The transitions are linked via arcs and the two places *Mechanism Response* and *Agent Choices*. These serve to exchange data in the form of dynamic tokens.

Mechanism Model (MM)

The mechanism model represents the resource allocation rule of the automated planning system, i.e. the rule that maps the situation of the game and the choices of the agents to a certain resource allocation with associated individual utilities for the agents. The structure, complexity, and properties of a specific mechanism model vary widely depending on the application domain.

A common property of the mechanism is that the response function will generally be deterministic. In the design of computerized information systems in general and planning systems for ATM-problems in particular, it is often adequate to model the world (the system surrounding) as a non-deterministic system. However, the planning system itself is almost always constructed as deterministic system. This design decision is driven by the desire to answer any given system input (e.g. perceived traffic situation) with the singular best-known system-output (e.g. planning solution). It also avoids unnecessary complexity in the design and analysis of the planning system.

Where the behavior of the mechanism model includes an important share of human decision making (i.e. a human operator substantially influences and controls the planning system and selects planning solutions) and where this operator behavior has to be considered as non-deterministic, this property can also be modeled. The inclusion of non-determinism in the mechanism model does not present a problem from the modeling point of view in the DAVIC framework. As pointed out in chapter 4, non-determinism is intuitively supported by Petri nets. However, non-determinism in the mechanism model adds additional complexity to the later stages of analysis, where appropriate decision-theoretic and game-theoretic solution concepts have to be applied.

Note that in order to compare different mechanism variants or parameters, alternative mechanism and resource allocation algorithms $MM_1....MM_n$ can be specified on the mechanism model subpage and be evaluated separately during the analysis phase. This is further treated in section 5.3 and demonstrated in section 6.9 on the application example.

Agent Behavior Model (AM)

The agent behavior model specifies the degrees of freedom for distributed agents when interacting with the mechanism center. In contrast to the mechanism model, the agent behavior model is thus intrinsically a non-deterministic model. It specifies the choices agents have for any given situation and given initial conditions and defines the finite strategy set $S_i = \{1, 2, ..., n_i\}$ which each player i may choose from. The initial conditions themselves are defined in the scenario specification (see also following paragraph 'Scenario Specification').

Note that if an agent has no choices in the context of a certain application scenario, he could as a matter of fact be excluded from the analysis with decision-theoretic or game-theoretic criteria. Decision theory and game theory are by their very nature concepts of how we use our freedom to make decisions [Hans05].

Further note that if none of the agents has any choices and degrees of freedom within the context of the application, the potential state space of a system collapses to a singular system trajectory. In this case the system output can be predicted easily by simple simulation of the system.

In order to compare different assumptions on agents' degrees of freedom, alternative agent behavior models $AM_1...AM_n$ can be specified on the AM subpage. These can then be activated separately during the analysis phase, due to the modular structure of the model.

Scenario Specification (Sc)

Scenario specifications Sc present initializations of the fixed net structure of mechanisms model and agent behavior model with dynamic data. The scenario determines

the initial marking of the net (see chapter 4).

Through the initialization of the net structure, the CPN model becomes executable. The data values of the scenario determine how states of the model may evolve within the model constraints.

The model structure in conjunction with a scenario initialization provides a full but implicit representation of all system states and outcomes which the combined system can reach. The implicit representation is made explicit by computing the system state space (see section 5.2.3).

Usually it is necessary to work not with a singular scenario specification, but with a comprehensive set of potential system scenarios $SC = \{Sc_1, Sc_2, ..., Sc_n\}$. The scenario set SC should cover the whole range of operational situations discovered during initial research on the application layer. The scenario set may be generated either by a systematic variation of certain scenario parameters identified on the application layer. Alternatively, e.g. Monte Carlo techniques may be used to generate random scenario sets within certain boundaries. In any case, the specified scenario set SC should be representative of the more informal operational scenarios described on the application layer.

5.2.3. State Space

State space generation

Based on the definition of the above system model SM, a state space SS is computed which represents all possible executions of the model that is analyzed. This means that all reachable states and state changes which can be generated by the interaction of the different agents (agent behavioral model AM) with the mechanism (mechanism model MM) are explored. The state space is represented as a directed graph (digraph). The nodes represent system states and arcs represent occurring events (players' choices, and mechanism response) [Jens97a] (see section 4.3.2).

Note that the resulting state space depends on

1. the mechanism model variant MM_i,
2. the agent behavior model variant AM_j, and
3. the specific scenario Sc_k.

Thus, when system properties have to be analyzed for m mechanism model variants MM_i and a agent behavioral model variants AM_j over s different scenarios Sc_k, a total number $numss = m \cdot a \cdot s$ of distinct state spaces SS has to be computed and analyzed.

State space representation

Figure 5.3 shows an abstract representation of a state space SS_t, which illustrates the interaction of agents with a mechanism. This state space is derived from the model presented in Figure 5.2 for the case of a mechanism variant MM_i, an agent behavioral model variant AM_j and a specific scenario Sc_k. In the state space graph, the nodes beginning with the letter *pn* represent states where one or several agents can choose between different options and the system mechanism is waiting for this input for further computations (player choice nodes). The nodes beginning with the letter *sn* represent states where the mechanism has to respond to the players' inputs, and the players are waiting for the response of the mechanism to proceed with their subsequent decisions. (In the given example, multiple agents can act simultaneously during several stages of the game, interrupted by the response of the planning system. The graph thus presents a case of a multistage game).

Definition 5.2.1 *A **Player Node** pn is defined as a node where at least one of the players has an non-empty set of available choices and the mechanism model is waiting for a choice to be taken.*

Definition 5.2.2 *A **System Node** sn is defined as a node where a response of the resource allocation mechanism is due before any of the players can make any new choice or system input.*

Depending on the specific model, the state space may between the player nodes contain one or several intermediate states resulting from the computations of the planning system. These system nodes are usually not of interest from a decision-theoretic point of view, due to the deterministic nature of the mechanism. Consequently the nodes which only have a single output arc may be omitted from the state space and replaced by adequate replacement arcs as indicated in the figure by black arrows spanning multiple gray arcs.

For complex systems the graphic presentation and visualization of full state spaces will often not be feasible due to the high number of states. Visualizing fragments of a state space can however be a very effective way of analyzing the markings reachable within a small number of steps from a given marking [Jens07].

State space analysis

State spaces make it possible to verify, i.e., prove in the mathematical sense of the word that the model possesses a certain formally specified property.

In the context of the DAVIC framework, the state space provides the basis for proving that the system outcomes desired by the system designer at the same time satisfy certain decision-theoretic criteria. This makes it plausible that the desired emergent

system performance will actually be realized in practice. Therefore, both the desired and the predicted behavior have to be formulated as mathematical expressions. This is realized in the respective components Agent-Rationality Concepts and System-Optimality Concepts of the DAVIC framework (see section 5.2.5 and section 5.2.6) on the theoretical basis introduced in chapter 3.

Note that with respect to the decision-theoretic and game-theoretic formalizations introduced in section 3.2, the state space defines the assumed underlying reality of the decision situation and game. However, the state space alone does not define the game itself. It still lacks important assumptions on

- what is defined as 'utility' and what is the 'payoff' for the different physical outcomes,
- what information on others players' moves and preferences is available, and
- which assumptions on agent rationality are made.

In particular this means that the class of the game which is chosen for this analysis (e.g. decision analysis, normal form game, extensive form game) is not determined by the state space itself. In fact, different decision-theoretic formalizations can be derived from one and the same state space.

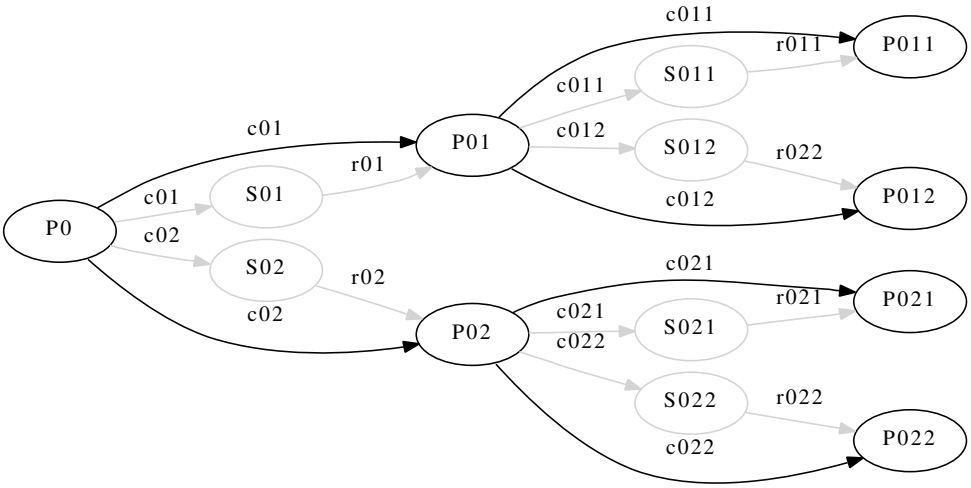

Figure 5.3.: Simple state space graph with alternating player- and system-turns. Determinstic mechanism response (light grey) can be omitted for game-theoretic analysis.

5.2.4. State Space Interfaces

Interfaces are needed to decouple the implementations of decision-theoretic- and rationality concepts from the structure and type of state space produced by a specific

system model. While the structure- and data representation of the components Application Layer, Model Layer, and State Space inevitably depends on the application, the components discussed below (section 5.2.5 to 5.2.8) should, in the interest of reusability of the framework, be as independent as possible of the specific application layer.

This decoupling is provided by a collection of interfaces, which map the concrete state space structure and the datatypes of an application to abstract constructs which are commonly used for the description of decision-theoretic- and optimization problems.

Two types of state space interfaces can be distinguished in the context of the DAVIC framework: Agent-Rationality Interfaces (ARI) and System-Optimality Interfaces (SOI). These are indicated to the left and right side of the state space component in Figure 5.1.

The purpose of Agent-Rationality Interfaces (ARIs) is to map the application dependent state space structure to a formalized and abstract decision problem. In order to do so, ARIs

- establish a formal definition of utility from the individual agents' perspective for each terminal state of an interaction, and
- define the availability of information for each player at each intermediate node of the game (if any),
  - according to the desired class of game, and
  - for the solution concepts which will be employed.

The purpose of System Optimality Interfaces (SOIs) is to map the application-dependent state space structure to a formalized optimization problem. SOIs support the system designer's point of view. In order to do so, SOIs

- establish a formal definition of utility from the system perspective for each terminal state of an interaction, and
- provide access to specific properties of agents' behavior which are of interest for the system designer.

All interfaces to the state space (ARIs and SOIs) shall be defined to be independent of the detailed implementation of the planning mechanism or agent behavior. Therefore, they should preferably access the marking of the model places *mechanism response* and *agent choices* over which the mechanism model and agent model interact (see Figure 5.2), and not the inner model parts. For some systems it may however be unavoidable to access the inner agent behavioral model (AM) places e.g. to obtain direct behavioral measures. Even if this is the case, the interface definition shall be constant over a certain design space in question.

Note that the relation between interfaces and state space is a n-1 relationship. Different formalizations of a decision problem may be implemented on the same state space

structure, that is 1...k ARIs and 1...l SOIs can be used as part of one optimization process.

Also, several different utility functions may be defined on the same state space. That way, different utility mappings of the type ($u : X \to \mathbb{R}$) from physical process outcomes (e.g. arrival time of an aircraft, fuel consumption) to an assumed system of abstract utilities for an actor k are tested on the model.

5.2.5. Agent-Rationality Concepts

This component implements solution concepts for decision problems as discussed in section 3.1. A solution concept is a formal rule for predicting rational agent behavior when interacting with the mechanism. A solver Sl within the DAVIC framework is an implementation of a specific solution concept.

In order to obtain a formal representation of the respective decision problem, the solver accesses the state space through an ARI-interface.

The solvers provided by the Agent-Rationality Component are implemented independently of a specific game and system definition. This independence facilitates re-usability on different applications as well as easy changes and re-engineering of the mechanism without modifying the solvers. In some cases, however, changes on the system may affect the interface, as discussed above.

A number of different solvers $Sl_1 ... Sl_n$ can work all on the same state space (and potentially on the same state space interface), and treat the game according to different assumptions on actor rationality, information- availability and other constraints. This allows a comparison of predicted outcomes and predicted system performance under different assumptions about the operational environment in which the system will work.

The selection of a set of specific solvers and their explanatory power and informative value depends on the application context and operational surroundings for the mechanism as defined in the application layer.

Notably, different solvers may treat the application as a simultaneous or sequential game and as a decision-analysis or game-theoretic problem, according to the assumptions on information availability and rationality assumptions for agents. No changes to the model or the state space interfaces are necessary to accomplish multiple investigations under varying assumptions.

The outcome produced by each solver Sl_{AR} is a set of nodes A_{AR} representing rational solutions according to the given agent-rationality concept (see section 3.2), and for the given model and scenario. Each of the nodes represents a certain strategy profile sp_{AR} with one strategy for each participating agent.

5.2.6. System-Optimality Concepts

This component implements System-Optimality concepts as discussed in section 3.3.

A System-Optimality concept *OC* is a formal rule stating the designer's point of view on how the game shall be played in order to realize the intended emergent system performance.

To compute the necessary measures of system performance and planning quality, the functions implemented in this component access the state space through an SOI-interface.

Within the framework, the following classes of system-optimality concepts can be distinguished.

Joint agent-utility

Concepts in this category aim at the maximization of 'social welfare'. Social welfare is here defined as the sum of utilities of the agents participating in the decision problem and game. As discussed and defined in section 3.3, two typical concepts in this category are Hicks optimality and Pareto optimality.

The implementation of the system-optimality component for these concepts yields the nodes in the state space which satisfy the respective conditions of Hicks optimality and Pareto optimality. Note that the social welfare realized as the outcome of the decision problem does explicitly *not* consider the interest of any third parties or stakeholders. These stakeholders may be influenced by the outcome of the decision problem but cannot actively influence the decision themselves. The organization in charge of establishing a specific resource allocation mechanism is potentially included in this group of stakeholders in the sense that it cannot promote a particular outcome once the mechanism is established.

Agent behavioral measures

For many applications, criteria can be defined which specify desirable agent behavior directly, independent of the payoffs which individual agents receive through their behavior. Applying such criteria can also be interpreted as measuring the payoff for the mechanism center. To some extent, such criteria alleviate the problem discussed for the Joint Agent Utility Concepts (i.e. neglection of third party interests) and demand certain behavior within the boundaries of the degrees of freedom (DOF) of the system. Agent behavioral measures may be necessary in order to ensure the safe operation of a process, the acceptability of operations to human operators in the process or the fulfillment of other operational feasibility criteria. The outcome from each criterion is a set of nodes, containing all outcomes where agents comply with a specified behavior.

Planning system quality measures

For many control and planning applications, the planning system itself during the planning process computes a number of quality measures over different candidate planning solutions. Often the very same quality measures which the planning system employs can be used for the post-hoc evaluation of the system optimality view. The underlying assumption is that the planning logic itself adequately represents the preferences of the system designer for a certain class of outcomes. Thus, the same criteria can be used to measure if the outcomes produced by the agents' decisions realize a high system- planning quality in terms of these measures. Note that if it was assumed that the planning-system quality measures did not represent the preferences of the designer, considering the mechanism for implementation would be a contradiction in the first place.

The typical outcome of this type of criterion is

- a set of nodes which realizes the maximum quality level in terms of the internal quality functions of a planning system, or
- a set of nodes which satisfies a certain predefined minimum quality level.

The outcome of each OC is a set of nodes S_{OC} representing desirable strategy profiles SP_{OC} which should be played from the designer's perspective.

5.2.7. Analysis Layer

Functions on the Analysis Layer build upon the computation of the different node sets A_{AR} and S_{OC} provided by the Agent-Rationality Component and the System-Optimality Component. Their task is to analyze the interrelation and meaning of these node sets mainly by performing the following three tasks:

1. Node set integration, i.e. building intersections and relative complements of sets
2. Establishing formal proofs of logical set relations and guaranteed set properties
3. Quantitative evaluation of set sizes and set properties

The Analysis Layer thus practically realizes the core tasks of the DAVIC framework: to contrast rational agent behavior with system designers' desired behavior and to support conclusions about incentive-compatibility of a mechanism.

Node set integration

Inputs of the Analysis Component are the node sets defined on the state space by the Agent-Rationality- and System-Optimality Component. These original node sets are termed *primary sets* in the context of the DAVIC framework. From these primary sets a number of *secondary sets* can be derived. In this process set operations such

as intersections, relative complement functions, and subsets play an important role. The definitions are provided in the following.

Definition 5.2.3 ***Subset*** *For any given sets A and B, A is a subset of B if every member of A is also a member of B. Notation: If a set A is a subset of B, this is written* $A \subseteq B$.

Definition 5.2.4 ***Intersection*** *For any given sets A and B, the intersection is the set of all elements in A and B that are in both sets. Notation: The intersection of A and B is written as* $A \cap B$*, thus* $x \in A \cap B$ *if and only if* $x \in A$ *and* $x \in B$*.*

Definition 5.2.5 ***Relative complement*** *For any given sets A and B, the relative complement of A in B is the set of elements in B, but not in A. Notation:* $x \in B \setminus A = \{x \in B | x \notin A\}$*.*

These secondary sets support the provision of information on

- how far the predicted outcomes of a decision problem coincide with the outcomes desired by the system designer,
- how far the system-optimality view is consistent in itself (different design requirements should have a common intersection of their node sets, otherwise the full list of requirements cannot be fulfilled, regardless of the eventual agent behavior), or
- how far the different agent-rationality concepts lead to similar solutions, or if the predicted outcome depends critically on the exact conditions of information availability and assumptions on agents' rationality.

The analysis of this information is conducted by constructing formal proofs and quantitative analysis criteria, as described in the following.

Formal proofs

Formal proofs provide evidence that the mechanism possesses certain desirable properties or does not possess specific undesirable properties [Jens07]. The ability to perform such formal proofs constitutes a core capability of the DAVIC framework. On an abstract level, two classes of proofs can be distinguished:

- Proofs regarding logical relations of primary and secondary node sets
- Proofs regarding characteristics & properties of set members

With view to the types and semantic meaning of the sets which are included in the proof there are also two different options:

- *Incentive compatibility* A first class of proofs analyzes the characteristics of both Agent Rationality Sets A_{AR} and System-Optimality Sets S_{OC} and the relation between them. This class of proof is regularly used for the main purposes of this framework, in order to establish incentive-compatibility of a mechanism. It provides evidence that the system outcomes desired by the system designer are compatible with rational (selfish) behavior of agents participating in the decision problem.
- *Inner consistency* A second class of proof considers either only agent-rationality sets A_{AR} or only system- optimality sets S_{OC} and their inner relations and characteristics. These are employed to analyze the inner consistency of the two perspectives.

Logical relations between agent-rationality sets & system-optimality sets

- *Subset proof* This proof provides evidence that each predicted solution of a game is at the same time a member of the node set desired by the mechanism designer. Formally this is achieved by a subset proof of the type $A_{AR} \subseteq S_{OC}$, where A_{AR} represents predicted outcomes of a game, and S_{OC} represents desired outcomes. The *interpretation* of a successful subset proof is that rational behavior of agents will automatically guarantee the implementation states desired by the designer. Acting against the system designer's expectations would be strictly irrational for at least one of the participating agents (see Figure 5.4 left).
- *Intersection proof* This proof provides evidence that a non-empty intersection of the predicted solution of a game and the design goals exists. The proof concept is thus realized by a proof of the type $A_{AR} \cap S_{OC} \neq \varnothing$. The *interpretation* of a successful intersection proof is that it is not irrational for participating agents to behave in the way expected by the mechanism designer, in other words, it exists at least one solution which is both rational and desirable. However, there may be other equally rational choices for participating agents and some of these might be undesirable from the system designer's point of view (see Figure 5.4 right).

The difference between the subset proof and the intersection proof is that for the latter $A_{AR} \subseteq S_{OC}$ does not necessarily hold. The intersection proof is thus denoted as a *weak proof*, the stronger subset proof includes the weaker intersection proof.

Logical relations within agent-rationality sets or system-optimality sets

- *System-Optimality Intersection* A special class of intersection proof can defined by an intersection of System-Optimality sets: This is used to assure that there is an overlap between the the node sets desired from a system's point of view (e.g.

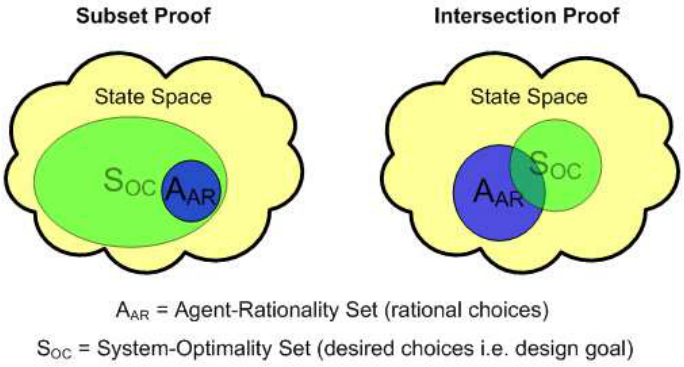

Figure 5.4.: Subset proof (left) demonstrates that rational agent behavior will guarantee the realization of the design goal. Intersection proof (right) demonstrates that rational agent behavior *may* realize the design goal, but some undesirable outcomes are also rational.

joint-agent-utilility criteria, behavioral criteria). Note that if the intersection of the different System-Optimality sets $S_{OC1} \cap S_{OC2} \cap ... \cap S_{OCn}$ turns out to be the empty set $\emptyset$, the design objectives are conflicting objectives by themselves and therefore cannot be realized simultaneously. This will then be true regardless of the fact that equilibria for agents' choices may lie within one of the different optimality sets.

- *Agent-Rationality Intersection* Analogously to System-Optimality Intersection all Agent-Rationality Sets can be intersected to build $A_{AR_{tot}} = A_{AR1} \cap A_{AR2} \cap ... \cap A_{ARn}$ in order to figure out if there is an overlap between concepts. If an overlap exists, or if the sets are even identical, this is an indication that the assumed rationality concepts (and related to that rationality constraints and informational situations of real actors) may not be critical. If, on the other hand, no overlap exists, the selection of the rationality concept and assumptions on information availability and rationality of actors in the real world may be crucial. In this case, controlling information availability in the operational environment may also be a means of engineering specific outcomes of the decision problem into the mechanism.

Property proofs

Property proofs are used to provide evidence of minimum or maximum set sizes or to prove guaranteed properties of all nodes belonging to a set.

The first variant is used to show that an intersection of two sets is sufficiently large, or that the relative complement of two sets may be sufficiently small to be accepted.

The second variant may be used to show that a specified minimum quality level in terms of the planning system's quality rating functions is always reached or that

all nodes of a set satisfy a certain minimum utility level e.g. in terms of joint or individual agents' utility.

Property proofs can be applied both to basic sets computed by the agent-rationality- or system-optimality components or on secondary sets derived from them.

Formulation and implementation of proofs

The procedure for formulating and implementing proofs on the basis of the computed state space of the CPN model is in most cases straightforward. The comprehensive library of state space query functions coming with CPN Tools and the modular structure of the DAVIC framework support this task.

The proofs regarding logical set relations can be formulated and checked by simple operations once the respective primary and secondary sets have been computed. They show that one set is a subset of another or that the intersection of two sets is not empty.

For the property proofs the condition is formulated as a predicate function on an individual node. This predicate function is then checked for all nodes of a set by one of CPN Tools standard state space functions, e.g. *PredAll* (see [Jens06]).

Both procedures return a Boolean value indicating whether the proof has failed or held for the combination of the specific mechanism variant and the scenario. If necessary, a list of counterexamples, i.e. nodes for which the proof has failed can also be computed[1].

Quantitative analysis

Based on the former computation of primary and secondary sets, a quantitative analysis is carried out. This analysis calculates the sizes of sets and characteristic variables for certain sets. A quantitative analyis is particularly informative where a more general (stronger) proof of desired properties fails (see formal proof above). In this case the consequences of agents behaving other than desired by the mechanism designer have to be quantified. The following perspectives may be taken by a quantitative analysis:

- *System cost analysis* The costs for the system are analyzed when an equilibrium (rational choices) is implemented that does not lead to the design goal. Typical cost measures could be overall loss of process time, system capacity, behavioral measures, planning system quality values, and joint agent utility. Thus this type of analysis aims at quantifying the cost difference for the case that an

[1] Note that in terms of the above distinction between proofs on logical set relations or set properties, it is generally possible to formulate semantically identical proofs on the basis of both approaches e. g. by prior computation of appropriate secondary sets.

outcome from the node set $N_1 = A_{AR} \setminus S_{OC}$ is realized instead of another outcome from the set $N_2 = A_{AR} \cap S_{OC}$. Also the size of the relative complement $N_1 = A_{AR} \setminus S_{OC}$ with regard to $N_2 = A_{AR} \cap S_{OC}$ is of interest, as it is related to the likelihood that an outcome from N_1 might be realized instead of an outcome of N_2.

- *Agent cost analysis* The costs for each agent are analyzed, assuming a system-optimal behavior would be realized by the agent which is not an equilibrium (and thus no rational choice for the agent). Typically, these costs could be the individual loss of time and individual utility measures. The quantification of such costs gives an indication of how hard it might be to convince agents of acting in the way desired by the designer. Thus this type of analysis aims at quantifying the cost difference if an outcome from a node set $N_1 = S_{OC} \setminus A_{AR}$ is realized, instead of another outcome from set $N_2 = A_{AR}$ [2].

5.2.8. Synthesis Layer

The synthesis layer provides functions to synthesize the different forms of analysis results from individual sources and bring them into a format and representation which facilitates decisions on the acceptance, rejection, or further development of a mechanism.

Regarding the type of representation, two graphical approaches can be distinguished. These consist of either a structure-dependent representation of results in the graph structure of the original decision problem (game tree representation) or a structure-independent representation of quantitative results in standard diagrams.

Additionally, these results are exported in textual form to data files, which can be imported into statistical programs or spreadsheet calculations for further analysis.

Structure-dependent representation

For this approach, analysis results are graphically represented in the state space graph of the game. The main advantage of this approach is that it supports the understanding of the decision-theoretic perspective as well as the system-optimality perspective by a graphical representation of the results in relation to the structure of the original decision problem. The following types of results can be represented within the graph:

- Agent-rational states / agent-rational strategy (rational choices of agents)

[2] Note that for game-theoretic problems the costs for an individual or even all agents may be negative (resulting in a factual win-win). This means that all individual agents and the overall system would be better off with a design goal outcome from S_{OC} than with an equilibrium state from A_{AR}. A decision problem which possesses such an incentive structure is called a *social dilemma*.

- System-optimal states / system-optimal strategies (behavior desired by designer)
- System states where design goal and agent-rational strategies do or do not coincide
- States where specific proofs hold or fail
- Quality criteria or behavioral measures for each state of the state space

As discussed above, a specific state space is always a product of a system model variant SM_i and one specific scenario specification Sc_j for initialization of that system. Consequently, the representation of the listed analysis results (agent-rational states, system-optimal states) also refers to that particular state space. Interpretations cannot usually be generalized directly over a set of different scenario specifications, as state space structure and analysis results will vary.

Taking this fact into account, the structure-dependent representation of results directly in the game tree is well suited to gain a deeper understanding of a particular example case and learn e.g. why an individual proof failed for a particular situation. It is less suited to represent results of average performance.

Also, for many decision problems of high complexity the size of the full graph will grow quickly. Visual inspection of the graph structure might only be practical for local areas of the graph.

Structure-independent representation

For this approach, analysis results are represented graphically but independent of the tree structure of a particular game. An important advantage of this approach is that it supports the evaluation of a mechanism and its performance over a comprehensive set of scenarios, e. g. in terms of average, minimum and maximum performance. Also, these predicted performance differences (e.g. in terms of processing times, planning quality) across different mechanism variants can be compared directly.

In order to represent the results, different types of standard diagrams are used such as bar-charts, surface charts etc. On the one hand, these diagrams allow a much more compact representation of results than the tree representation and a better quantification of differences. They also permit to mark directly the mechanism variants and parametrizations which perform optimal under a given evaluation criterion. On the other hand, some information on the underlying decision problem is lost in the structure-independent representation.

5.3. Design Space Search with DAVIC Framework

The DAVIC framework is built to facilitate a systematic optimization and search over a defined design space DS. By this procedure, the mechanism variant with the best

properties from a decision-theoretic and emergent-system performance point-of-view is identified. The considered mechanism variants which constitute the design space DS can be

- different parametrizations of structurally identical mechanisms (e.g. with different weightings of optimization functions in the planning system), or
- structurally distinct mechanisms with algorithmic differences in the planning logic of mechanism model MM, agent behavioral model AM and interaction protocol IP.

In Figure 5.5 a principled representation of a design space is shown with five structurally different mechanism variants which are tested for ten different parametrizations each. This leads to a total of 50 variants in this case.

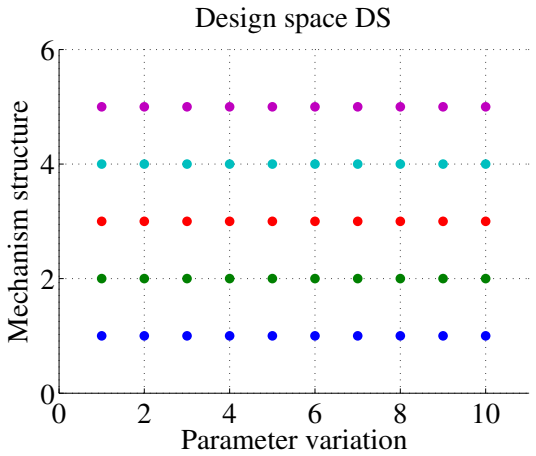

Figure 5.5.: Abstract mechanism design space DS

During the design space search, each mechanism variant $MM \in DS$ is confronted with the same set of scenario specifications SC. State spaces are computed for each combination of the MM with all individual scenarios Sc. This approach is indicated in Figure 5.6 by the set of distinct state spaces, one for each combination of MM and Sc. The different state spaces SS are accessed via the same interface and processed by the same analysis methods. Each state space is analyzed using the criteria discussed within the scope of the analysis layer. The results are then synthesized and represented with functionalities from the synthesis layer.

The purpose and result of the design space search is to select a mechanism variant which is optimal according to a specified criterion C. The selection criterium C can be of different types and complexity, such as to maximize

- the percentage of cases in which an individual formal proof holds over the given scenario set or
- a quality value as a weighted sum of different formal proofs and quantitative evaluation criteria or

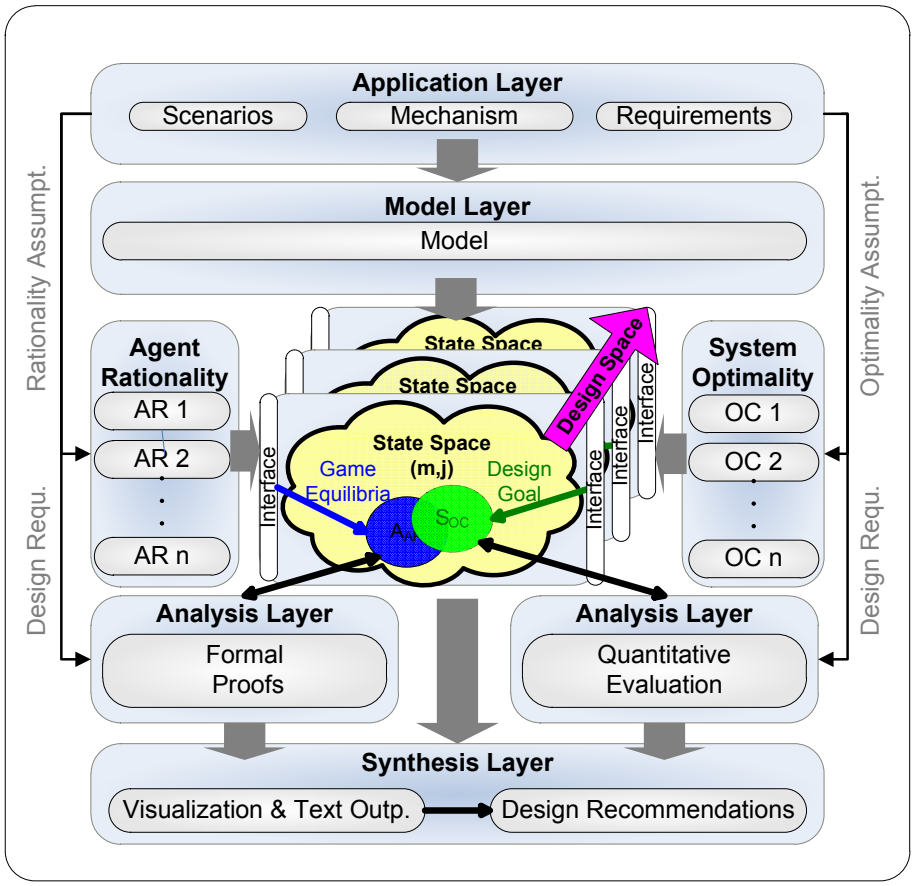

Figure 5.6.: Searching design space DS for the optimal mechanism variant with regard to the specified evaluation criterium C

- one criterion for a given minimum guaranteed level of another criterion.

The optimization procedure supported by the DAVIC framework is a one-step search and optimization procedure in which results are computed systematically for specified finite sets of mechanisms and scenarios representing the entire design space DS. Following this computation, the mechanism variant M_{opt} which best fulfills a specified criterion (see above) is selected, such that

$$M_{opt} = \arg\max_{M \in DS} C_o v(M). \tag{5.1}$$

5.4. Summary

In chapter 5 the DAVIC framework for the design and verification of incentive-compatible ATM systems has been presented.

The DAVIC framework serves to analyze distributed agent interactions. It enables the confrontation of agents' view (i.e. individual agents' interests) with system's view (i.e. system designer's interest). A defining feature of DAVIC is the tight integration of net-based modeling techniques for distributed systems with decision- theoretic and system-theoretic analysis methods which are combined within a common engineering framework.

Through the modular framework structure with its eight dedicated components, the framework supports the re-usability of models, analysis functions, and interfaces. The DAVIC framework with all components working together allows the identification of mechanism variants from a given design space which best fit the criterion of incentive-compatibility. It supports the design and identification of mechanisms in which emergent system behavior (as an interactive result of individual agents' rational choices) fits best the desired system behavior.

The services and performance of DAVIC are closely linked to the main theory underlying the framework:

- Decision theory and game theory provide the mathematical basis for predicting rational agents' choices.
- Established modeling tools from engineering provide methods for the simulation of distributed systems and (in the case of Coloured Petri Nets) also for the calculation of solution spaces.

In addition to existing work in the individual fields of decision theory and net-based modeling, the integrated DAVIC framework offers the following new contribution in particular:

In decision theory, it is usually assumed that the mechanism and available action space are already given. In DAVIC, while taking into account decision-theoretic criteria, the usage of a powerful modeling approach allows the systematic engineering of new mechanisms. Further, the modeling of potential system behavior and exploration of action spaces with established engineering methods does so far not support arguments on the rationality of agent behavior when interacting with the system. Modeling alone gives no answer on how to narrow down the solution space to the likely outcomes and emergent behavior. Therefore a tight integration of both modeling and decision theory is necessary.

The DAVIC framework, as presented, allows to check with affordable effort how mechanism changes affect solution spaces and how this in turn affects predicted agent behavior and emergent system performance. DAVIC thereby facilitates the design of incentive-compatible ATM systems within a structured approach. Thus, the

development of the DAVIC framework makes an important engineering contribution to closing the method gap outlined in chapter 2.

At the same time, the underlying constitutional theory of the framework (decision theory, Petri Net modeling) and the chosen framework structure itself result in certain challenges and current limitations for the approach:

- The *theoretical assumptions of decision theory and game theory* have been discussed in chapter 3. These assumptions can be hard to fulfill for some real-world applications and can present limitations in predicting actual agent behavior. The limitations have to be considered also when these solvers are employed within the framework. A potential benefit should be that the framework is open to the plug-in of future refined solvers and interfaces. These can be needed to reflect specific characteristics of a system or future theoretical developments.
- *The basics of the modeling approach* employed in the current implementation of the framework, (namely Coloured Petri Nets and CPN tools) and state space analysis haven been introduced in chapter 4. Notably CPN is a discrete modeling technique. This may represent a significant limitation depending on the system under consideration. Also the size of state spaces which can be computed is limited.
  While replacement by another modeling technique is not impossible and simulators for hybrid Petri nets are available, the state space computation for hybrid (discrete-continuous) systems is an open issue, and even more so is the treatment of such 'action spaces' with decision-theoretic means. This suggests that the limitation of the discrete modeling approach might be fairly persistent.
- The *search algorithm* for the identification of the optimal mechanism is currently a one-pass selection from a predefined design space. No special intelligence is implemented so far for the creation of new mechanisms (i.e. the iterative, constructive refinement of mechanisms, genetic algorithms etc.). This approach may turn out to be insufficient where the theoretical design space (of different mechanisms) becomes very large. Then not all combinations of mechanism variants and scenarios can be exhaustively tested. In this case, an enhancement with an algorithm with iterative refinement of the solution space (e.g. genetic algorithm for creation of mechanism candidates) should be considered.

All identified current limitations present starting points for further development of the framework and the analysis approach on the methodological level. Despite the challenges, a wide class of ATM problems is assumed to exist, where the application of the framework can benefit the system development today. To demonstrate the practical use of the framework on an example it will be applied to the problem of an arrival management planning mechanism in chapter 6. In chapter 7 the results of the study will be presented.

6. Realization of DAVIC Framework for Cooperative Arrival Management

This chapter applies the framework for the Design and Verification of Incentive-Compatible ATM Systems (DAVIC) developed in chapter 5 to a class of potential future mechanisms for arrival management. Arrival management is the process of organizing converging streams of traffic to an airport in order to safely and efficiently use a set of common and constrained airport resources such as airspace and runway.

The novelty and difference of the investigated mechanisms with regard to today's practice lies in an extended data exchange and collaboration between airborne actors (aircraft) and ground-based actors (ATC and planning systems) in order to determine the arrival sequence. This leads to a situation in which

a) in the positive case, an optimized arrival sequence with more efficient flight profiles is built. Future arrival management will consider individual aircraft capabilities, flight-situation, preferences, and other sequence constraints better than this is possible today.

b) as a negative side-effect, the mechanism might become more vulnerable to selfish behavior and manipulation, due to the new choices and possibilities of interaction. If some actors experience that they may profit from uncooperative behavior, the effects could destabilize the sequence and reduce overall efficiency.

It is a central purpose of the presented analysis with the DAVIC framework to prove that cooperative interaction with the ground-based planning system (the arrival manager AMAN) is actually in the individual aircraft's best interest. Specifically, a timely and truthful submission of aircraft estimated times of arrival times to the central planning system should be *rational*. In order to show that cooperative and truthful submission of estimates is rational, the abstract definition of incentive-compatibility from chapter 3 is adapted and applied to the specific arrival-management problem under consideration. Where incentive-compatibility cannot be guaranteed unconditionally for all scenarios and situations, the potential negative effects should be analyzed and quantified.

The overall analysis set up for the mechanism-design problem in this chapter considers different variants from a class of mechanisms in order to find out which solution within the design space satisfies the above requirements best. Trade-offs are identified which may exist between the different variants. Specifically, different planning functions to stabilize the sequence are tested and their impact in terms of incentive-compatibility is investigated.

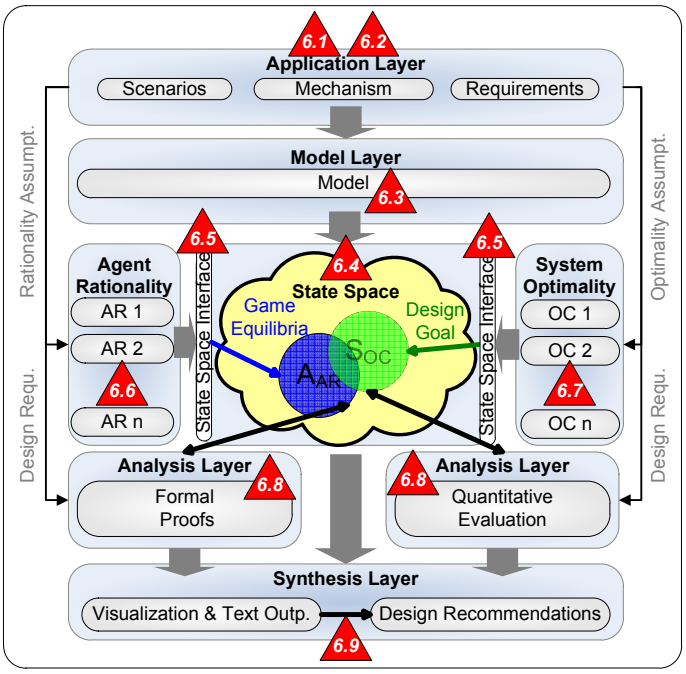

Figure 6.1.: Framework application - correspondence between framework components and subsections

Chapter structure

This chapter 6 is structured according to the component-structure of the generic DAVIC framework presented in chapter 5. One by one, the meaning of each framework component in the context of the arrival management application is explained. After the entire system and subject of the analysis has been established, the results are presented in chapter 7.

Figure 6.1 contains an overview showing in which section the application of each framework component is discussed. The numbers in the red triangles are section numbers, meant to support the quick look-up of specific component details.

6.1. Application and Context

The description of the DAVIC application-layer component is divided into two parts:

- Part I in this section 6.1 presents a brief general overview of arrival management.
- Part II in the following section 6.2 will present the specific example case to be analyzed.

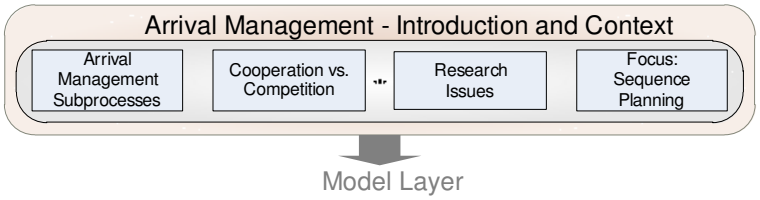

Figure 6.2.: DAVIC Application Layer - Part I

The most important topics of part I are shown in Figure 6.2. These are the subprocesses of arrival management (with a focus on the subprocess sequence-planning considered later), cooperative and competitive elements in arrival management, and an outline of development trends and research issues.

6.1.1. Introduction to Arrival Management

Managing the arrival traffic to a highly frequented airport is one of the most challenging tasks of air traffic control. Much of the difficulty arises from the fact, that arrival management requires a synchronization of formerly uncoordinated streams of aircraft to use a limited set of shared airport resources as efficiently as possible. These shared resources include constrained airspace, common waypoints, and scarce runway time.

Generic subprocesses Looking at today's situation, the exact procedures applied at a specific airport vary widely. Geographic constraints, airspace specifities, the user community and traffic mix play important roles. However, regardless of the exact procedures, the following four generic subproblems have to be solved:

- *Sequencing,* i.e. establishing a favorable arrival sequence (sequence of aircraft) according to a number of optimization criteria,
- *Metering,* i.e. to determine for each individual aircraft appropriate target times over (TTO) certain points (fixes) of the respective arrival route,
- *Trajectory generation,* which means finding an efficient and conflict-free route for each aircraft from its current position to the runway threshold, and also
- *Clearance generation,* i.e. to generate instructions which should be given to an aircraft in order to lead it along the route as planned.

A simplified view of the arrival management including the main processes and information flow is illustrated in Figure 6.3 (adapted from [Ober06]). In practice, additional dependencies and feedback loops may exist which are not represented in this Figure.

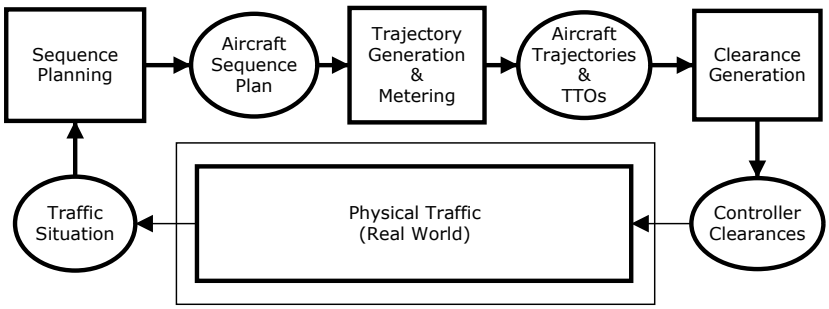

Figure 6.3.: Simplified view of arrival planning process

Research Due to the difficulty of the arrival management process, and its criticality in terms of throughput, flight efficiency and noise, arrival management procedures and support systems (AMANs) have been under investigation for a number of decades.

Pioneering work on tool support for arrival management goes back to Völckers [Volc90]) with the development of the COMPAS system at DLR and to Erzberger with the work on Approach Spacing tools at NASA/USA (e.g. [Neum90]).

Nowadays, basic arrival management systems (AMANs) are in operational use at many major airports. At the same time, further improvements and functional extensions to AMANs are an active R&D topic both in the SESAR and NEXTGEN programs. In SESAR for example the operational SWP 5.6 (Queue Management in TMA and Enroute) and the technical projects 10.9.1 (Integration of Queue Management) and 10.9.2 (Multiple Airport Arrival/Departure Management) work in this field.

Recently the German Aerospace Center (DLR) has conducted intensive research on air-ground integrated arrival management (e.g.[Korn06]). New concepts for organizing and guiding traffic in the Extended Terminal Maneuvering Area (E-TMA) were investigated e.g. in the FAGI project to realize more fuel-efficient and noise-reduced approaches (e.g. [Ober08c]) with the arrival manager 4D-CARMA (e.g. [Hann08]).

Arrival management usually has a number of goals, including improvements of runway throughput, individual flight efficiency and reductions of noise and emissions. These goals are in many ways conflicting. To reach and balance them, the need for effective automated planning tools supporting the human air traffic controller in the realization of this challenging task becomes increasingly important.

6.1.2. Cooperative Mechanisms in Arrival Management

Future technological and operational changes

A number of technological and operational changes are expected for the future development of arrival management. These changes will have considerable impact on

the way in which actors work together:

- *Higher automation levels* More powerful Arrival Managers (AMANs) will bring higher automation levels to all outlined subprocesses (sequence planning, metering, trajectory planning etc.). This will change some of the planning decisions from human decisions into automated planning with human intervention only by exception.
- *Integration of air- and ground systems (AMAN and FMS)* A direct coupling- and data exchange will be established between the ground-based AMAN and the aircraft's Flight Management System (FMS) via a digital datalink.
- *Higher information availability* Information availability will improve in the air and on the ground through CDM and SWIM applications. Additional systems will appear to support this, such as Cockpit Displays of Traffic Information (CDTIs).
- *Bidirectional trajectory negotiation* Trajectories will no longer be unilaterally imposed on the aircraft by the air traffic controller. Rather, the trajectory will be the result of a bidirectional negotiation between air- and ground (system). Therein, the user's (aircraft's) preferences and performance data will explicitly be taken into account.
- *4D-trajectory contract* The negotiation will result in a 4D-trajectory contract. The contract is an agreement on which trajectory the aircraft is to follow and at which spatial point is to be at a certain point in time.

Cooperative vs. competitive elements

The outlined changes make arrival management a more cooperative process in principle. Their explicit aim is to establish improved coordination to better satisfy airspace-user needs. As a side effect, however, the very same changes may also increase the potential for strategic interaction. Manipulation may become possible as is pointed out in the following.

- *Influence of higher automation levels* Human decision-making is generally not deterministic. Automated planning systems, however, tend to be deterministic systems. For identical traffic situations and interaction scenarios, such systems produce the same system response every time. This increases transparency. It may also support the risk of manipulation, as it can make profits of uncooperative behavior calculable and reproducible.
- *Influence of better information availability* Information availability regarding system state and traffic situation is the basis for situated behavior. Situated behavior can again be either cooperative or uncooperative in nature. The result can, but must not necessarily, be advantageous for global system performance.

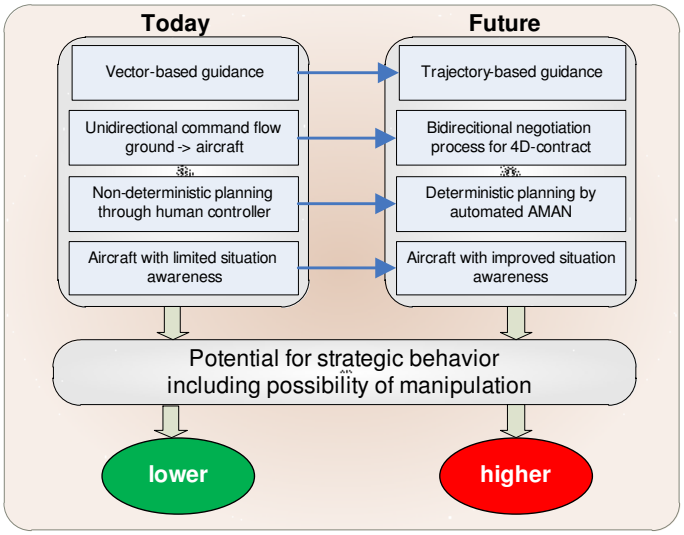

Figure 6.4.: Potential for strategic interaction in arrival management - today vs. future

- *Influence of available choices in negotation* Future arrival management processes will offer new choices to the aircraft. As part of the trajectory negotiation influence on the process by the aircraft is explicitly intended. But this influence and the choices offered may be used to the good or the bad of overall system performance.
- *Influence of communication mode* The change from voice communication to data link has many effects, but one side-effect is the anonymization of communication. It is to be assumed that the anonymization of communication via datalink might relax social norms of cooperative behavior and make strategic actions for the optimization of individual results more likely, as personal contacts are weakened [Hans00].

Figure 6.4 illustrates the development from today's to future arrival management procedures and summarizes their effect on the potential for strategic interaction and optimization or manipulation by an individual aircraft.

Given the strong economic competition between airlines (see chapter 2), it seems likely that also the potential for uncooperative exploitation will be tested.

The consequence is that in the engineering process careful attention must be given to the incentives set by any new planning system or procedure. Wherever possible it must be shown that what is understood as 'cooperative behavior' of agents from the point of view of system design is also in the individual agents' own best interest. The individual goals of the arriving aircraft (and flightcrew) and the system goals for the arrival management system must be aligned.

6.1.3. Sequence Planning

For the purpose of the analysis of arrival management mechanisms in this work, the focus is on the sequence-planning subprocess of arrival management. The sequence-planning process is the natural starting point for the modeling of the behavior of an arrival management mechanism. Sequence planning precedes the other planning processes (trajectory generation and metering) and is the most critical one for all actors in terms of punctuality, costs, and fuel consumption.

Subprocesses The task of sequence planning can be divided into four subtasks, which a planning mechanism has to consider. These are sequence generation, sequence evaluation, sequence selection, and sequence implementation (see Figure 6.5) which are explained in the following:

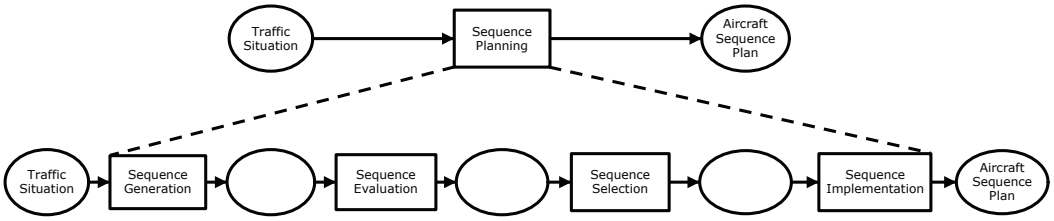

Figure 6.5.: Subprocesses of sequence planning [Ober08b]

- *Sequence Generation* The sequence-generation step produces a set of candidate sequences for a number of aircraft within a defined timeframe. The set includes all possible permutations of aircraft within a certain time-frame. Note that for a set of $NoOfAc$ aircraft one will get $NoOfCseq = NoOfAc!$ candidate sequences. For each candidate sequence, the respective arrival times for each aircraft are computed. These arrival times have to conform with the required temporal and spatial separation between consecutive aircraft.
- *Sequence Evaluation* The sequence-evaluation step computes a quality value for each candidate. This quality value reflects the desirability and feasibility of practically realizing that candidate sequence from an operational point of view. The criteria for this evaluation are implemented in 'rating functions', rating different performance- and quality issues about the sequence.
- *Sequence Selection* Based on the quality values, the one sequence to be implemented in practice is picked in the sequence selection-step.
- *Sequence Implementation* After the sequence has been selected it becomes an input to the trajectory generation and metering process (see Figure 6.3). The planning result is translated into appropriate instructions (clearances) to traffic participants in order to implement the sequence.

6.2. Example Case: Sequence Planning Mechanism for Merging Arrival Traffic

This section introduces the specific example application and planning mechanism which is analyzed in this and the following chapter. In terms of the DAVIC (Design and Verification of Incentive Compatible Systems) framework, the description forms part II of the *Application Layer* component (see section 5.2.1). As indicated in Figure 6.6 it includes

- the specific operational scenario of a merging situation for arrival traffic,
- the planning mechanisms considered,
- the initial traffic situations and events which the system has to react to,
- the choices available to agents (aircraft) in reaction to these events, and
- the designer's expectations regarding agents' reactions and the requirements concerning emergent system performance.

In the first step the system outline is formulated as an informal textual description. This description is detailed and formalized further as an executable Coloured Petri Net (CPN) Model in the following section 6.3.

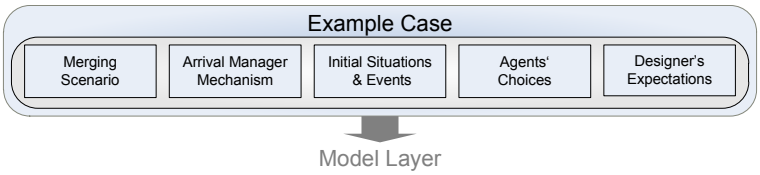

Figure 6.6.: DAVIC Application Layer - Part II

6.2.1. Traffic Merging Situation and Planning Mechanism

In the example-case studied arrival-traffic is merged to a common Merging Point MP located on the approach. An automated sequence-planning mechanism is applied in order to establish appropriate time separation between aircraft at the merging point. The sequence- and time- planning is based on aircraft inputs and air-ground data exchange. Aircraft submit their Earliest Arrival Times (ETAs) for this waypoint. The sequence-planning is continuously updated in reaction to potential disturbances of the system.

Merging situation & route structure The merging situation is depicted schematically in Figure 6.7 (left).

- In the operational scenario a merging situation is assumed with n routes to the common merging point MP.

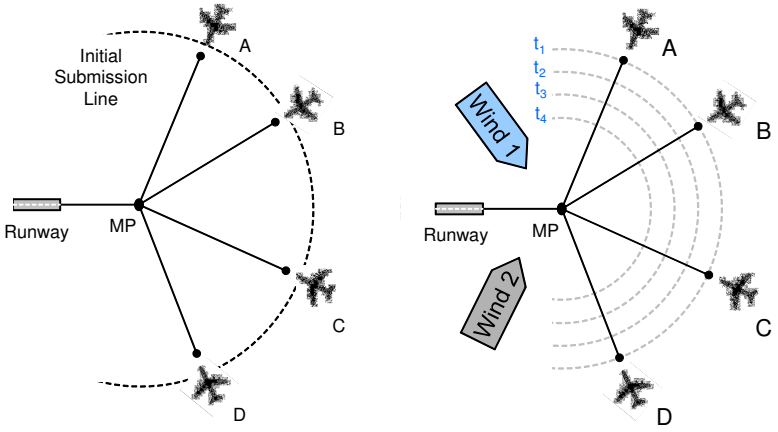

Figure 6.7.: Merging situation and route structure

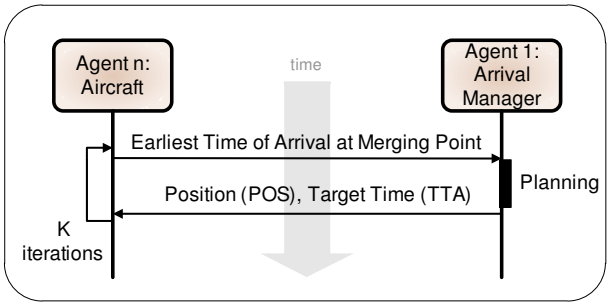

Figure 6.8.: Simple air/ground communication protocol

- At the merging point a time separation $tsep \geqslant 75\,s$ between two consecutive aircraft must be established for safety reasons and to avoid wake turbulence.
- All n routes are considered to be 'independent'. That means that sufficient separation at MP is thought to imply sufficient separation at any time before MP, regardless of potential differences in aircraft approach speed.
- After passing the merging point MP all aircraft fly a common route segment and use a common runway resource.

Planning mechanism The outline of the planning mechanism to establish a sequence for MP is as follows (See Figure 6.7 and sequence diagram in Figure 6.8):

- Upon entering a geographic horizon around the airport (Figure 6.7 (left),'Initial Submission Line'), each aircraft is required to submit its 'Earliest Time of Arrival' (ETA) for the Merging Point MP.
- Based on the available ETAs, the AMAN's sequence planner establishes a favorable and conflict-free sequence and Target Times of Arrival (TTAs) for MP.

- Its target time TTA and sequence position POS is returned to each aircraft (with TTA$\geqslant$ETA).
- The planning is updated repeatedly as new information becomes available.

Rating functions For sequence evaluation, the AMAN uses two kinds of rating functions. These enforce the two (partially competing) sequence planner's goals:

- *Optimization* For each aircraft a TTA should be planned as close as possible to its ETA (i.e. as early as possible) while maintaining necessary separation.
- *Stabilization* A high degree of stability in the sequence should be assured in order to limit the control effort of air-traffic controllers and the need for air-ground communication and renegotiation.

The optimization function is based only on the latest set of available ETAs. The stabilization functions however are recursive and have the preceding planning-cycles state as inputs, too. This adds to the complexity of the process as it makes the sequence result dependent on the entire history of submissions rather than just on the set of currently valid ETAs.

Formal definitions of the specific functions which are designed to realize these goals of optimization and stabilization are introduced in section 6.3.2 and Appendix A.2.

Traffic scenario Traffic scenarios are defined in terms of

1. initial situation,
2. scenario events (i.e. disturbances), and
3. agents' choices in reacting to these events.

These three elements build the basis of the decision problem which the agents face in each scenario.

Initial situation The initial situation for each scenario is as follows:

- Each traffic scenarios considers a set of aircraft approaching on different routes. These aircraft have to be sequenced for the waypoint MP. For the following exmple, the scenarios contain a set of four aircraft A,B,C,D.
- For the initial situation it is assumed that all aircraft are sequenced in the order of their earliest time of arrival, that is $ETA_1 \leqslant ETA_2 \leqslant ETA_3 ... ETA_{n-1} \leqslant ETA_n$. By convention, the first aircraft in the sequence is denominated A the second B and so on. Thus, $ETA_A \leqslant ETA_B \leqslant ETA_C \leqslant ETA_D$ holds for the initial sequence S_{ini} .

- All ETAs in the initial sequence are considered to lie within an time range TR. If $TR < (n-1) * tsep$, a resource conflict arises. It means that at least one aircraft will receive a TTA later than its ETA. This aircraft is delayed. This aircraft would benefit from switching position with the preceding aircraft. These two aircraft have a conflict of interest.

Scenario event & disturbances The sequence planning does not always stay stable with regard to the initial sequence:

- Different events and disturbances such as unexpected weather changes, imprecise predictions of flight performance or other operational incidents can cause changes to the Earliest Time of Arrival (ETA) of an aircraft.
- In the case of such ETA changes, TTAs may become unreachable, or an earlier TTA may become desirable for an aircraft.
- A planned sequence may have to be recalculated and the sequence positions and TTAs of aircraft in the sequence may have to be updated.

In the analysis performed in this thesis the modeled disturbances are caused by wind changes. When a wind shift occurs, the aircraft approaching the airport from different directions are either delayed or accelerated with regard to their initial planning. This effect is reflected in a shift of the Earliest Times of Arrival (ETAs) of aircraft over the Merging Point MP. The difference between the ETA in the initial situation and the new ETA related to the scenario event is termed $etashift$=$(ETA_k - ETA_{k-1})$.

AMAN system response In reaction to the modified inputs for ETA as submitted by agents and the characteristics of the planning system, the planning system AMAN may replan the sequence. The replanning can represent modifications of the position and target times of all or individual aircraft in the sequence. This will be favorable for some aircraft and unfavorable for others. Practically, the changes will translate into variations of fuel cost and delay with regard to the original schedule.

Efficiency/stability trade-off An important property of the planning system is associated with the need for an efficiency/stability trade-off. Some smaller changes in ETAs may not justify sequence switches. The consequence is that the resulting sequence will not always be sorted in the order of ascending ETAs.

Agents' choice The agents' choices in the scenario consist in timing the submission of the ETA-shift:

- The correction of ETAs (submission of ETA-shift) itself is mandatory. It cannot finally be avoided by the aircraft.
- However, the aircraft has considerable choice in the timing of the submission to the planning system.

- The update may be submitted as soon as detected or
- it may also be delayed for some time and submitted later.

The choice of timing determines the chance of an aircraft to be the first, second, third or fourth aircraft to submit the modification. Thus the aircraft influences the submission order of ETAs. Due to the recursive nature of the planning process, the submission-order affects the sequence outcome. That way, timing the submission of ETA-shifts becomes a strategic instrument for an agent.

Observability An important issue is the lack of observability of the submission behavior of the aircraft (and of the flightcrew, if it is assumed that the submission requires manual intervention). It is in practice very difficult to control for an external observer on the ground (i.e. the planning system) if an update is submitted as soon as possible under the given circumstances. This is so, because important knowledge about the operational situation is encapsulated by the aircraft. Only the flightcrew can know the earliest time at which the submission was operationally feasible. The lack of observability makes it difficult to impose early submission as a mandatory rule because such a rule could hardly be enforced in practice.

Designer's expectations & requirements The system designer's expectation is that the aircraft should correct the time prediction ETA as soon as the change is discovered (that is, as soon as practically feasible for the flightcrew). The timely submission of updates is required in order to keep the planning system up to date. It guarantees that all plans produced by the planning system are practically feasible and efficient. Also, the earlier modifications to the plan are made, the lower the cost for all agents[1].

Since the expected behavior is difficult to enforce by regulation, it is desirable that this behavior be in a selfish agent's own best interest. That way, the agent can be expected to act in the desired way where this is possible.

Design & verification objectives The purpose of the analysis is to verify if the designer's expectations are compatible with the incentive-structures established by the planning system's dynamic behavior. Specifically, the analysis serves to decide

1. if the available choice (timing of submission) affects the outcome, or if the outcome is generally independent of agents' choice,
2. if it is actually in each agent's (rational) interest to announce necessary modifications of the Earliest Time of Arrival as soon as possible,

[1] While not explicitly modeled in the following, at some point TTA changes can no longer be complied to by speed changes, and more complex lateral re-routing may have to be found.

3. if the behavior expected by the designer (early submission) will also be rewarded by the planning system and will be experienced as favorable by the agent, and
4. if it is possible to systematically influence or manipulate the response of the planning system in a strategic manner.

The answers to these questions are to be considered in the definition and selection of the system design.

6.3. Model of Cooperative Arrival Management

This section develops the formal description of the introduced arrival management process and merging situation as an executable Coloured Petri Net (CPN) Model. The CPN Model constitutes the 'Model Layer' Component of the DAVIC Framework (see section 5.2.2). The description includes

- the modeled design space DS of planning mechanisms,
- the traffic scenarios Sc which are analyzed,
- the actors' choices in the traffic scenarios,
- the interaction protocol IP between agents and the planning mechanism.

The discussion is divided into three parts (see Figure 6.9):

1. In subsection 6.3.1 the overall model structure and the functions of its individual modules are introduced. Here, the discussion focuses on the structurally constant, *static model* parts, i.e. the parts that stay the same for all different mechanism variants within the design space (Figure 6.9 left).
2. In subsection 6.3.2 the rating functions and parameters for their relative weighting are discussed. These elements represent the *variable model parts* and allow the definition of alternative mechanisms spanning the design space DS in the following (Figure 6.9 middle).
3. In subsection 6.3.3 the generation of the *scenario set* is discussed. The scenario set is used to initialize the model in the following analysis when the properties and performance of different mechanisms from the design space DS are compared (Figure 6.9 right).

Technically and with regard to state-space analysis methods, the CPN model is based on previous work, namely on [Ober06, Ober08b]. Therein, initial versions of a cooperative arrival planning models were presented. By extending this work, the model has been adapted to the specific merging situation and has been restructured for more efficient automated analysis within the DAVIC framework. Further, parts of the model have been outsourced into textual program descriptions. These changes allow a more efficient state space analysis and avoid creation of unnecessary nodes in the state space.

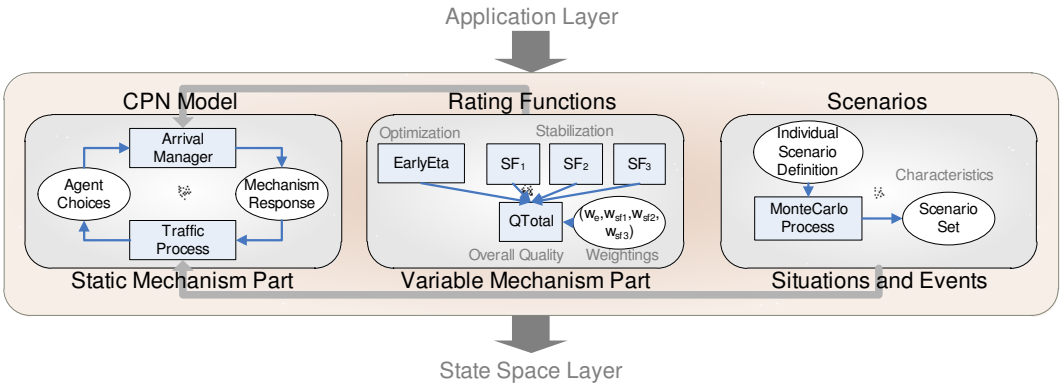

Figure 6.9.: DAVIC Model Layer

6.3.1. Coloured Petri Net Model

The CPN model used for the analysis consists of 4 pages organized in 3 hierarchical levels. All these pages derive from the *Initialization* page. The page hierarchy is depicted in Figure 6.10. With reference to the DAVIC framework terminology, the model pages can be devided into the three conceptual modules System Model (SM) Mechanism Model (MM) and Agent Model (AM).

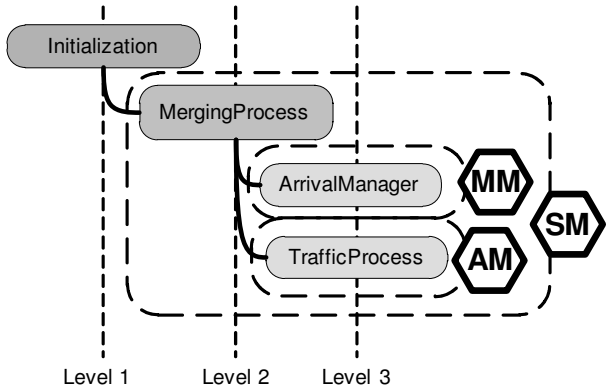

Figure 6.10.: CPN model - page hierarchy

In the following, all four pages are discussed. The discussion of each page is divided into a brief description of the page function and purpose as well as a more detailed implementation description. All non-standard colourset definitions used in the description are detailed in listing A.1 of the appendix.

In addition to model-functionalities, which are represented graphically, various processes rely on textual program parts and subroutines. These are referenced and included in the discussion where needed.

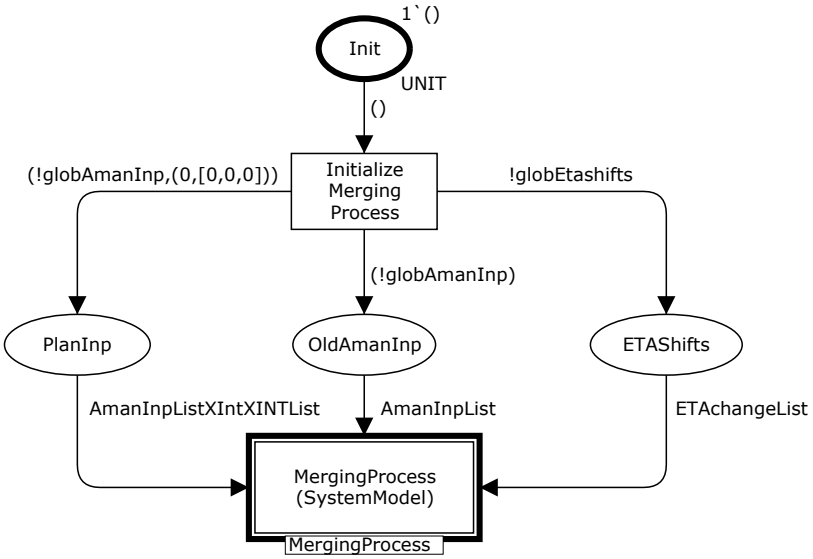

Figure 6.11.: Initialization page

Initialization Page (SM)

Function/purpose The *Initialization* page is depicted in Figure 6.11. Its purpose is to feed the model with scenario data at the beginning of the analysis.

Implementation The initialization page features the regular transition *InitializeMergingProcess* and the substitution transition *MergingProcess*.

The transition *InitializeMergingProcess* reads out two reference variables *globAmanInp* and *globEtashifts* on its output arcs. This loads a specific traffic situation (*globAmanInp*) and a set of choices available to the agents (*globEtaShifts*). The variables contain scenario information previously produced by the scenario generation process (see section 6.3.3). They are placed as tokens on the places *PlanInp*, *OldAmanInp* and *ETAshifts*.

The substitution transition *MergingProcess* and its subpages implement all processes concerned with the simulation and analysis of the merging situation introduced in section 6.2. This transition represents the entire System Model (SM) in terms of the DAVIC framework (see section 5.2.2).

The whole process of initializing the reference variables is controlled and executed by code structures from outside the CPN model which were developed in previous work, namely [Ober08a] and [Ober08b].

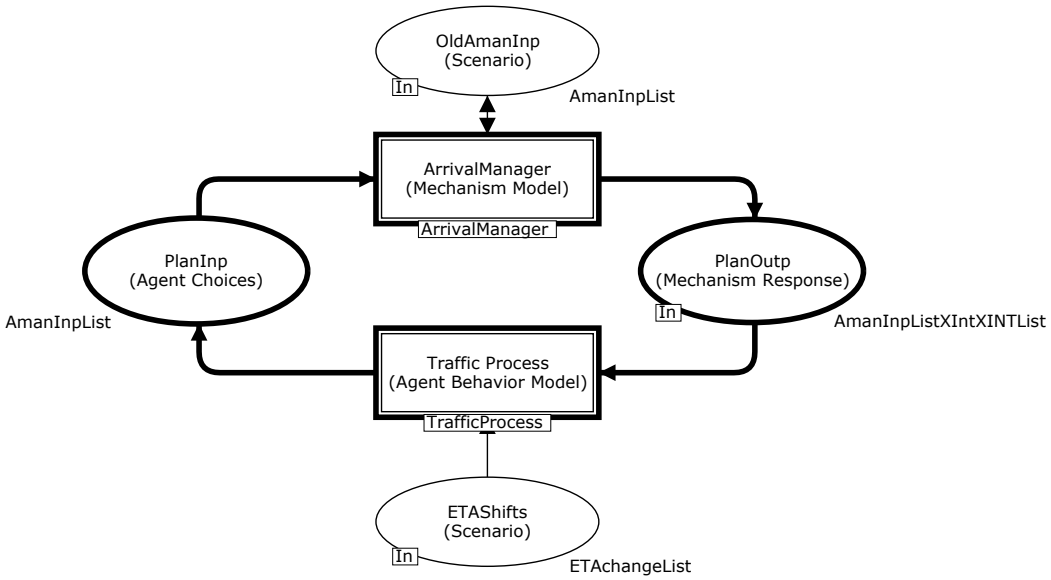

Figure 6.12.: MergingProcess page

MergingProcess Page (SM)

Function/purpose The *MergingProcess* page is depicted in Figure 6.12. In terms of the DAVIC framework structure, it implements the system model *SM*, consisting of the three different components Mechanism Model (*MM*), Agent Behavior Model (*AM*), and Interaction Protocol (*IP*) (compare section 5.2.2).

Implementation The upper substitution transition *ArrivalManager* encapsulates all processes related to mechanism model *MM,* specifically the resource allocation rule. The lower substitution transition *TrafficProcess* encapsulates possible agent behaviors and choices. These two model parts communicate via the places *PlanInp* and *PlanOutp.* Due to the closed-loop structure of the process, the PlanOutp to the ArrivalManager is the input to the TrafficProcess and Agent's choices. The other way round, the PlanInput to the ArrivalManager is the the traffic situation and choices made by the agents.

The datatypes and type of information exchanged between the two transitions are given in Table 6.1. Input- as well as output-situation are modeled in terms of the parameters Callsign (CS), Earliest Time of Arrival (ETA), Target Time of Arrival (TTA) and Aircraft Position in Sequence (POS). No geometric information or dynamic aircraft is used. Further, individual and aggregate quality ratings of the actual plan by the planning system are processed as the PlanOutp. Note that this quality information is used only for later analysis purposes on the state space. It is not accessed by the agent model *AM* to reason about the rationality of actions.

Place: PlanInp **Colourset:** AmanInpList	**Place:** PlanOutp **Colourset:** AmanInpListXIntXINTList
• Aircraft Callsigns CS • Earliest Times of Arrival ETA • Target Times of Arrival TTA • Aircraft Positions in Sequence POS	• Aircraft Callsigns CS • Earliest Times of Arrival ETA • Target Times of Arrival TTA • Aircraft Positions in Sequence POS • Total Quality value of Sequence assigned by Arrival Manager • List of individual quality values by respective rating functions

Table 6.1.: Input and Output coloursets of Places PlanInp and PlanOutp

Finally, two more places are visible on this model page. The place *OldAmanInp* is initialized with the initial traffic situation at the beginning of the simulation. For the execution of the scenario the marking of this place is then repeatedly updated. For each planning cycle k it represents the state of the traffic situation from the preceding planning cycle $k-1$.

The place ETAshifts is initialized with the choices available to actors. As the simulation progresses and *ETAshifts* are executed by aircraft, the set of remaining *ETAshifts* on the place is reduced until no more choices remain. This terminates the scenario after κ stages.

ArrivalManager Page (MM)

Function/purpose The purpose of the ArrivalManager page (Figure 6.13) is to implement the Mechanism Model (MM) and resource-allocation rule for the cooperative planning process. In line with the discussion in section 5.2.2, this resource-allocation rule is a deterministic process, which means that for the same system-state/inputs the output is always the same.

Implementation Via the place *PlanInp* information on the actual traffic situation and actual earliest times of arrival ETA is received for time k. On the place

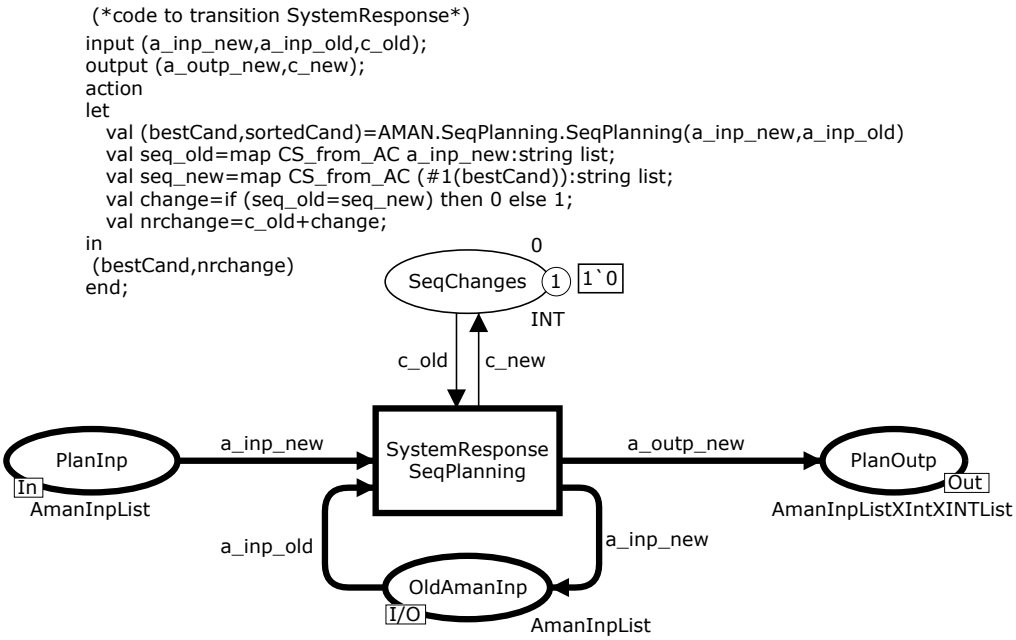

Figure 6.13.: ArrivalManager page

OldAmanInp, the planned sequence/traffic situation from the previous planning cycle *k-1* is still available to enable stability calculations.

When the transition *SystemResponse* fires, the execution of the function *AMAN.SeqPlanning.SeqPlanning* is triggered via the code segment connected with the transition. The code behind this function implements the four processes *Sequence Generation*, *Sequence Evaluation*, *Sequence Selection* and *Sequence Implementation* introduced in section 6.1.3. The result is the plan for a new sequence. This is issued as a token on the place *PlanOutp*. Further, if the new sequence is not identical to the old sequence, the value of the integer token on the place *SeqChanges* is incremented by one.

It is pointed out that the *ArrivalManager* page itself describes the input/output structure and datatypes of the process, it also fixates the recursive nature of the process. However, the fundamental issues of sequence-quality evaluation to determine which candidate sequence should be implemented is not described on the page itself. This evaluation is based on a set of rating functions called by the *AMAN.SeqPlanning.SeqPlanning*. These rating functions are detailed in section 6.3.2 and appendix A.2.

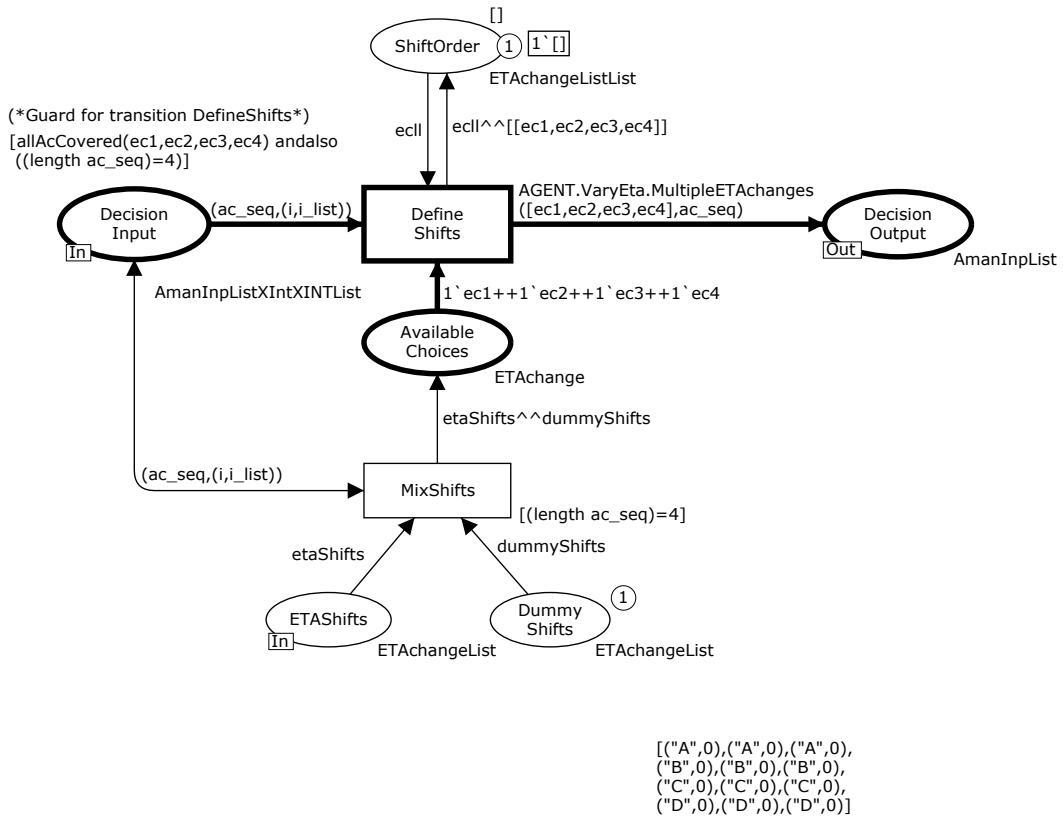

Figure 6.14.: TrafficProcess page

Traffic Process Page (AM)

Function/purpose The purpose of the *TrafficProcess* page (Figure 6.14) is to model the changes in the traffic situation and agents' choices in the interaction with the planning system. In terms of the DAVIC framework it implements the Agent Model (AM). Specifically, the page models agents' freedom in timing the submission of ETA-shifts. It applies the submitted modifications of the earliest time ETA to the actual traffic situation.

As has been described above (section 6.2), the size and number of the ETA-shifts themselves is determined by the scenario. The agent's freedom consists in the decision when (at what time, i.e. during which planning cycle) to submit the necessary changes.

In order to reflect agents' freedom of choice, this part of the model contains a non-determinstic process. This non-determinism is explored by the state space analysis technique, it causes the branches in the decision-tree and leads to different potential outcomes.

Implementation The *TrafficProcess* page contains two input places which are named *DecisionInput* and *ETAshifts*. The place *DecisionInput* receives a token with the actual state of the traffic situation from the superpage *MergingProcess*. The place *ETAshifts* receives a list of tokens defining the potential ETA-shift(s) for each aircraft according to the scenario S_{ini} which has been loaded. Further, for the place *DummyShifts* an ETAchange-list contained in the initial marking of the net. *DummyShift* means that the datatype/colorset of the tokens is *ETAchange*, but the value of the *etachange* is zero, and thus the change has no factual impact. However, for each planning cycle, either a DummyShift or a real shift is needed for each aircraft. This means that the number planning of cycles which can be simulated is determined by the number of real ETAshifts plus the number of DummyShifts available for each aircraft. When these have been consumed, no more planning cycles will be executable and the simulation stops.

Of the two transitions visible on the page, the first one to fire is always the *MixShifts* transition. This transition fires only once during the execution of the first planning cycle. The purpose of this transition is to merge the two separate *ETAchange* lists with real *ETAshifts* and *DummyShifts* into one multiset of *ETAchanges* on the place *AvailableChanges*.

The main functionality of the page lies in the transition *DefineShifts*. It fires once during each planning cycle. The transition binds a) the actual traffic situation *ac_seq* from the input place *DecisionInput*, and b) one *Etashift* for each aircraft in $ec1, ec2, ec3$ and $ec4$ from the place *availableChoices*. The function *allAcCovered* in the transition-guard ensures that exactly one ETA-shift token is consumed for each aircraft and $ec1$ to $ec4$ correspond to unique aircraft. On the output arc of the transition the function *Agent.VaryEta.MultipleETAchanges* is applied to the traffic situation. This substitutes the current ETA_x of aircraft x in the sequence by then new $ETA = ETA_x + ETAshift_x$. The result is sent to the place *DecisionOutput*.

Finally, on the place *ShiftOrder*, the order of all *ETAchanges* applied to the sequence is saved as a parameter for state- space queries in the later analysis.

Fixed aircraft number Regarding the process as implemented on this page, it is in exactly this formulation applicable only to scenarios with four aircraft. This is mainly due to the fact that the transition *DefineShifts* consumes as an input the the multiset $1'ec1++1'ec2++1'ec3++1'ec4$, which demands *ETAshifts* for exactly four different aircraft. Alternatively, a generalized *n-Actor* version of the process can be realized. However, for technical reasons, the generalized version has a more complex structure and consists of more transitions. It therefore causes more (unnecessary) nodes in the state space. Due to this, the more specific 4-Player variant is here preferred, putting computational efficiency above modeling economy. If necessary, an adaptation to variants with a different number of players is easily reached.

6.3.2. Rating Functions

The CPN model defines the outline of the merging process, the communication protocol used. It also determines how the process can be simulated and analyzed. To determine exactly which sequence should be implemented and preferred to the other sequence candidates, a set of rating functions is employed. The rating functions assign a scalar quality value between 0.0 and 1.0 to each potential sequence candidate. The candidate sequence with the highest quality value is selected and implemented (see section 6.1.3).

Generation of mechanism variants Regarding the relation of the CPN model and rating functions the following distinction is made:

- The graphical CPN model represents a constant structure for all mechanism compared in this work (see Figure 6.9, left).
- The rating functions on the other hand represent the variable mechanism elements (see Figure 6.9, middle).

The combination and parametrization of the rating functions will span the design space DS of mechanisms for the following experiments.

Individual rating functions Four individual rating functions are implemented:

- The function $EarlyETA$ is an optimization function that rates how close the assigned TTA of each aircraft comes to the submitted ETA of that aircraft.
- The function SF_1 rates the stability of the sequence in cycle k by evaluating potential differences in aircraft target times with regard to cycle $k-1$.
- The function SF_2 provides an alternative approach to rating sequence stability. It evaluates how many positions of the sequence in cycle k are still held by the same aircraft as in cycle $k-1$.
- The function SF_3 provides another alternative to rating the stability of the sequence. It evaluates the magnitude of sequence changes (in contrast to SF_2, which analyzes their number).

Aggregation to total quality rating The total quality value Q_{Total} is calculated as the weighted sum of the individual ratings from $EarlyEta$, SF_1, SF_2, and SF_3. The corresponding weights are w_e, w_{SF_1}, w_{SF_2} and w_{SF_3}.

The mathematical details and further explanations to all five functions $EarlyEta, SF_1$, SF_2, SF_3, and Q_{Total} can be found in Appendix A.2.

6.3.3. Monte Carlo Scenario Generation

In order to conduct a simulation and to verify properties of a candidate mechanism, scenario specifications are needed to initialize the fixed net structure of the CPN model presented in section 6.3 with dynamic data. On the application level, the scenarios confront the mechanisms with a concrete traffic situation. That way, a state space of available options and consequences can be computed and the efficiency of different mechanism variants can be compared. Since the dimensionality of the scenario space is too high even for a low number of aircraft to allow a complete set of all possible scenarios to be computed, a Monte Carlo process is chosen. This generates a randomized scenario set $SC = \{Sc_1 ... Sc_s\}$ representative of a range of practically possible traffic situations.

Definition of individual scenario Sc_i

The individual scenario specifications Sc used in this work consist of the elements

- initial traffic situation S_{ini}, represented by a list of aircraft $AC_1 ... AC_n$ each with
  - aircraft callsign CS,
  - initial earliest times of arrival ETA,
  - initial target time of arrival TTA,
  - initial sequence position POS, and
- scenario events E, represented by
  - modifications of the earliest time ES (ETA-shift) due to a weather event, one for each aircraft.

Monte Carlo Process A Monte Carlo process is used to generate and instantiate the individual scenario specifications Sc_i. Combinations of earliest times of arrival ETA and ETA-shifts ES are generated by a random process. The process of generating a scenario specification Sc_i is defined by the following five steps:

1. For a scenario Sc with a number of n aircraft, n time values $tv_1 ... tv_n$ are drawn from a discrete uniform distribution in the range $[ETB_{lower}, ETB_{upper}]$.
2. The time values $tv_1 ... tv_n$ are assigned to n earliest times of arrival $ETA_1 ... ETA_n$, so that $ETA_1 \leqslant ETA_2 .... \leqslant ETA_n$.
3. A set of n target times $TTA_1 ... TTA_n$ is computed, so that $TTA_i = max(ETA_i, TTA_{i-1} + t_{sep})$.
4. A list of n aircraft $AC_1 ... AC_n$ is computed, with $AC_i = \langle CS_i, ETA_i, TTA_i, POS_i \rangle$

5. By convention, callsigns CS_i are assigned to the sequence in ascending alphabetic order $CS_1 = A$, $CS_2 = B$...
6. Finally, for each aircraft AC one ETA-Shift is generated by a random process by drawing ES_i from a uniform distribution in the range $[ESB_{lower}, ESB_{upper}]$.

Definition of scenario set SC

With the described Monte Carlo process, a scenario set SC is defined for the analysis in chapter 7 with inputs/parameters as follows:

- The number of scenarios is $s = 500$.
- The number of aircraft is n =4.
- The separation is $t_{sep} = 75$s.
- The range of initial ETAs is between $ETB_{lower} = 0$ s and $ETB_{upper} = 150\,s$ (3 slots).
- The range of ETA-Shifts between $ESB_{lower} = -75\,s$ and $ESB_{upper} = 75\,s$ (1 slot).

The scenario set SC is saved in a text-file format and is later loaded into the CPN model for its analysis.

More detailed characteristics of the scenario set SC used for the following analysis can be found in appendix A.3. There, the exact distributions of the initial earliest times of arrival ETA, initial target times of arrival TTA and available ETA-shifts ES for each aircraft in the scenario set is found and commented further.

Discussion of scenario generation approach

Monte Carlo Approach A Monte Carlo approach is used for generation of the scenario set SC, in particular for the definition of initial traffic situations and scenario events. Monte Carlo processes are a common analysis technique for the numerical approximation of high-dimensional problems where closed analytical solutions are unknown or too computationally expensive and regular (non-stochastic) discretization of the input space may introduce systematic biases. In the presented case the following elements are considered to be statistically independent:

- the initial earliest times of arrival ETA of aircraft participating
- the direction from which they arrive
- how they are therefore affected by wind changes, i.e ETA-shifts

The described Monte Carlo process is thought to produce a valid and representative scenario set over the potential scenario space.

Randomized scenarios vs. complete state spaces With regard to the overall analysis approach, it is important to note that

- the scenario-generation process generates a randomized, arguably representative, but inherently *incomplete set* SC of individual scenario specifications Sc_i. That is, SC does explicitly not contain all possible individual scenarios Sc.
- the decision-theoretic analysis of a mechanism for an individual scenario Sc is a formal analysis of a *complete state space* for that scenario. That is, the state space does explicitly include all decision options possible within the modeled process for the individual scenario Sc_i. The properties of the mechanism for that scenario can be formally verified.

Implementation From an implementation point of view, the Monte Carlo process described is realized as a Coloured Petri Net using the same modeling technique as for the system- and mechanism model. The non-deterministic properties and firing rules of the CPN as well as dedicated random-distribution functions are used to implement the stochastic elements. The exact formulation and implementation details are omitted here in favor of the more compact textual description of the Monte Carlo process above.

6.4. State Space of Cooperative Arrival Management

This section desribes the State Space Layer Component of the DAVIC framework with the elements shown in Figure 6.15. Each state space SS_i is generated on the basis of the system model SM (section 6.3.1) and an individual scenario specification $Sc_i \in SC$ (section 6.3.3). For the creation of the state space, the automatic state space generation technique presented in section 4.3.4 is used. The most important structural properties of the state space graph for the application are introduced. Each individual state space SS_i will be accessed by other components through the state space interfaces for analysis purposes.

State space characteristics

Important characteristics of the state space are the following:

- *State space* The resulting state space is a directed tree $SS = (V, A)$ where
  - V is the set of nodes, and
  - A is a set of ordered pairs of nodes, called arcs.
- *Node types* The nodes in N are divided into three disjoint subsets which are
  - system nodes Sn (definition 5.2.2),

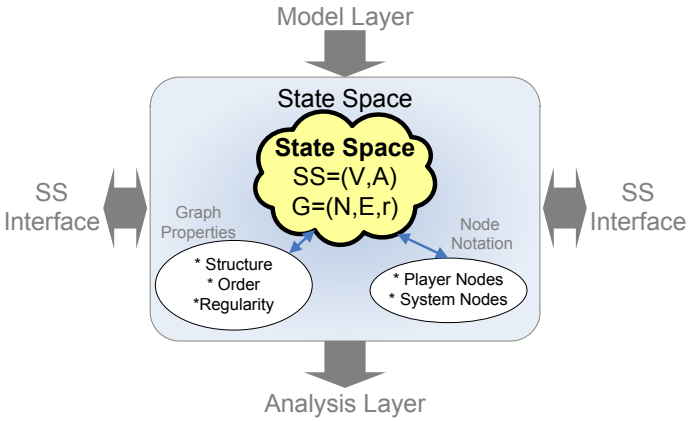

Figure 6.15.: DAVIC State Space Layer

- player nodes Pn (definition 5.2.1), and
- initialization nodes In.

Initialization nodes In represent the technical steps needed to load the scenarios into the CPN model. They have no meaning in terms of the decision-theoretic analysis of the application.

- *Gamegraph* Let the rooted graph $G = (N, E, r)$ be defined as a subgraph of SS so that
  - the set of nodes $N = Pn \cap Sn = N \setminus In$,
  - the set of edges E contains all edges from A which connect player nodes Pn and system nodes Sn, and
  - the root r is the first player node of G representing the situation S_{ini} of the traffic scenario (labeled $N4$).

  The graph G is called *gamegraph*.
- *Node type alternation* Player nodes Pn and system nodes Sn alternate when traversing G from the root to the leafs. The root and all leaves are player nodes.
- *Order of graph* The maximum order of the graph G is

$$\mid N \mid = 2 \cdot \left(\left(\sum_{s=1}^{c} s^n \right) + s^n \right) - 1,$$

  where c denotes the number of distinct choices of each of the the n players. Due to the alternation,
  - $\left(\sum_{s=1}^{c} s^n \right) + s^n$ of these are player nodes and

– $\left(\sum_{s=1}^{c} s^n\right) + s^n - 1$ are system nodes.

For the arrival management scenario with $n = 4$ players and $c = 4$ distinct choices for each player follows$| N |= 1119$.

- *Number of leaves* The maximum number of leaves $L(G)$ is c^n. For the arrival management scenario, this corresponds to $L(G) = 4^4 = 256$ leaves.
- *Order reduction* Some scenarios can contain ETA-Shifts of ES=0 for some actors, meaning that these actors have factually no choices. In this case, the number of players n is reduced so that $L(G)$ and $\mid N \mid$ are lower than the maximum value.
- *Irregularity* Graph G is irregular.
  - For player nodes Pn the indegree deg^- is one and the outdegree varies so that
  $$1 \leqslant deg^+ \leqslant \sum_{i=0}^{n} \left(\begin{array}{c} n \\ i \end{array} \right).$$
  - For system nodes Sn the indegree deg^- and outdegree $deg^+$is one.
- *Tree height and node depth* The height of G is $s \cdot 2$. The depth of all leaves is equal.
- *Stage* The notion of *stage* is defined for a node in Pn, so that $stage(pn) = depth(pn)/2 + 1$. This notation is useful for the purpose of the game-theoretic evaluations where the decision points in the tree are of primary importance.

Node properties and graphical notation

A fraction of an example state space is depicted in Figure 6.16. The following notation is used in this and all following Figures with game-tree representations:

System Nodes

System nodes are represented as oval or diamond shapes:

- *Oval shapes* represent system nodes Sn where the sequence remains stable. That means the sequence in preceeding and succeeding player nodes is identical.
- *Diamond shapes* represent system nodes Sn where the sequence switches. That means the sequence in preceding and succeeding player nodes differs.

The labels of the system nodes give information on agents' choices in the preceding step, that is potential changes in the traffic situation which the planning system then has to react to. If any ETA-shift was submitted during the execution of the traffic

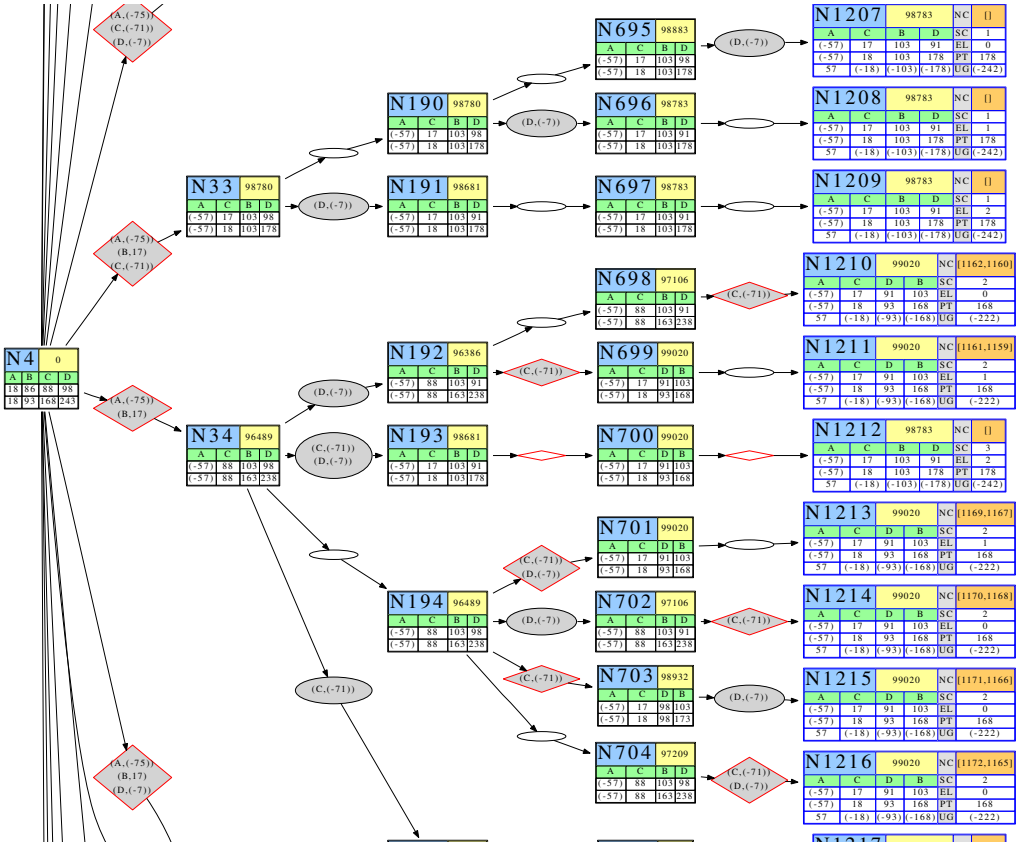

Figure 6.16.: Fraction of a state space for cooperative arrival management (For legend and detailed explanation of nodes see Figure 6.17)

process in the preceding planning cycle, the ETA-shift is indicated in the system node, e.g. $(D, -7)$ would state that aircraft D submitted an ETA-shift of -7.

Three examples of system nodes are presented in the left part of Figure 6.17:

a) A system node with ETA-shift but without sequence change

b) A system node with ETA-shift and sequence change

c) A system node without ETA-shift and without sequence change

Player Nodes

Player nodes Pn are represented as rectangular shapes. The node labels contain information on the system states simulated by the Coloured Petri Net. Two representations are used for leaf nodes and intermediate nodes. In the right part of the Figure the notation for a leaf node of a game tree G is represented, consisting of the following elements:

1. Node number
2. Quality value Q_{total} (equation A.12)
3. Nash Counterexamples (see definition 3.2.15)
4. Sequence of aircraft callsigns
5. Earliest Times of Arrival ETA in order of sequence
6. Target Times of Arrival TTA in order of sequence
7. Utility value U for each individual aircraft in order of sequence (see equation 6.1 below for the definition of the utility function).
8. Number of sequence changes C experienced on the path from the root node to this node (equation 6.9)
9. Number of $EmptyMovesLast$, i.e. iterations without ETA-shifts, counting backward from the leaf node
10. Total process time PT (sum of flight times of all aircraft, see definition 6.8)
11. Total utility value J (see equation 6.7).

In the middle part of the Figure 6.17 the label of an intermediate player node is depicted. The representation features only a reduced information set in comparison to the leaf node, for reasons of space and because some information elements are not meaningful for intermediate nodes. Apart from this, the same legend as listed for the leaf node applies.

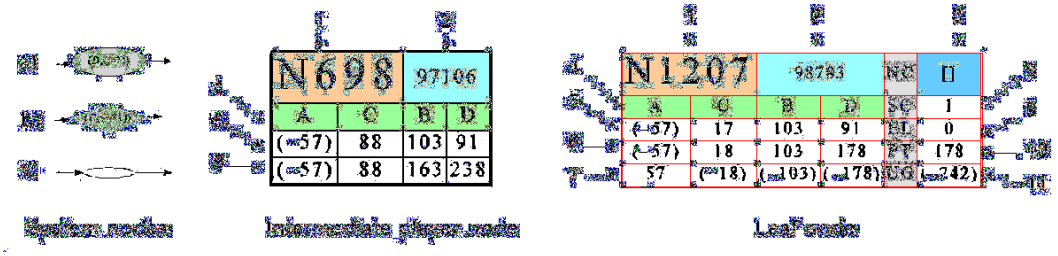

Figure 6.17.: Legend for state space nodes

6.5. Implemented State Space Interfaces

State space interfaces in DAVIC decouple the implementation of decision-theoretic and system-optimality analysis concepts from the specific structure of the application model. The framework distinguishes between agent-rationality interfaces (ARIs) and system-optimality interfaces (SOIs) (see section 5.2.4). In Figure 6.18 an overview is shown over the state space interfaces implemented for both types. The individual interfaces are discussed in the following.

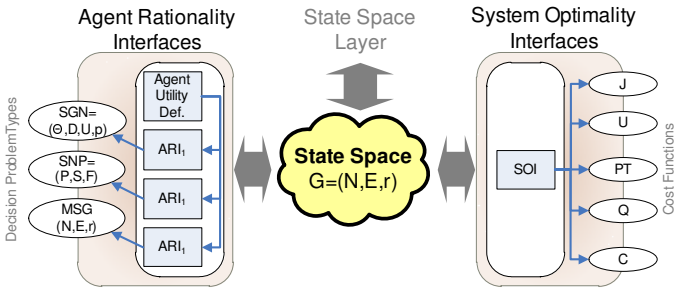

Figure 6.18.: DAVIC State Space Interfaces

6.5.1. Agent-Rationality Interface ARI

Agent-rationality interfaces ARIs map the state space graph G to a formal representation of a specific class of decision problem. In particular, an ARI establishes a definition of utility for individual players and defines information-availability for players during the stages of the game.

Three different ARIs are implemented for the analysis in this work. All three share a common definition of utility, which is discussed first. The ARIs differ, however, in their assumptions on information-availability and agent-rationality. These assumptions are pointed out separately for each ARI.

Individual agent-utility

The utility U of an outcome for an agent is determined by two elements which are

1. the target time TTA of the agent associated with that outcome, and
2. a reward by the mechanism center, depending on the agent's submission-behavior.

Formally, the utility is calculated as

$$U_{node,player} = -TTA_{node,player} + earlyMove_{node,player} * esr \tag{6.1}$$

with

$$earlyMove_{node,player} = c - a_{node,player} \tag{6.2}$$

where

$$
\begin{aligned}
node &= \text{state space node representing the outcome,} \\
player &= \text{the agent,} \\
c &= \text{the total number of potential submission slots,} \\
a &= \text{the slot chosen by } player \text{on the path to that outcome,} \\
esr &= \text{a constant early-submission reward parameter.}
\end{aligned}
$$

It is assumed that a mapping M from leaf nodes in G to outcomes and vice versa is known and unambiguous, and thus the terms *leaf nodes* and *outcomes* can be used interchangeably.

Applied interpretation of utility The first summand means that the earlier the TTA in the final sequence the better the utility for *player*.

The second summand establishes an external reward by the mechanism center in order to motivate agents to submit changes as early as possible. The earlier the agent submits the ETA-shift, the greater the reward by the system. The magnitude of the reward for early submission and its importance relative to influence of target time is scaled by the parameter *esr*. Operationally, the reward for early submission is assumed to be awarded as a monetary bonus (e.g. a reduced landing fee).

Note that the sequence position of the aircraft is assumed to have no direct impact on utility, but of course sequence position and target time TTA are closely interrelated.

Note that U, despite being a utility which agents try to maximize, will usually have values below zero in this formulation. However, only the relative order of utilities is relevant for the subsequent analysis, not their absolute values.

6.5.1.1. Simultaneous Game Against Nature ($\mathbf{ARI_1}$)

The formalization of ARI_1 corresponds to the set-up of a single player P acting against nature (section 3.2.1). The agent-rationality interface realizes a mapping

$$
\begin{aligned}
ARI_1 : G &\longmapsto SGN \\
\Leftrightarrow ARI_1 : (N, E, r) &\longmapsto (\ominus, D, U, p).
\end{aligned}
\tag{6.3}
$$

In this formulation N denotes nodes, E edges, and r the root of the state space graph G as described in section 6.4. These are mapped to a decision problem SGN with the elements outcome space Θ, decision space D, probability mass function p and utility function U (see section 3.2.1).

Interpretation In the applied sense of the cooperative arrival management problem the components of SGN have the following meaning:

- Decision space D defines all choices d available to player P, that is all options for P to submit Δeta (here: cycle 1...4).
- Outcome space Θ contains all possible combinations θ of other aircraft choices, that is all different histories in which aircraft other than P can submit their ETA-shifts. These choices are considered as 'moves of nature' from the perspective of P.
- Probability mass function $p(\theta \mid d)$ gives the probability that θ will take a certain value $\theta \in \Theta$ given that P chooses $d \in D$. For ARI_1 it is assumed that all other aircraft act randomly. It follows that $p = c/L(G)$.
- The utility function $U(d, \theta)$ specifies the payoff of the actor if he makes a decision d and the outcome is θ. Given a mapping $M : (d, \theta) \longmapsto node$, this is defined as $U_{player,node}$ according to equation 6.1.

Information availability/access The formalization of the application treats it as a single-stage decision problem. No information is available to player P during the several stages of the arrival management process with regard to the choices previously taken by other actors.

In the applied sense, this means assuming an aircraft which holds knowledge on

1. the resource allocation mechanism, and
2. the other actors' payoff functions, but
3. during the approach has no up-to-date information on the state of the world which it could use to determine choices.

The information used for the construction of this decision problem is thus primarily based on the leaf nodes of the tree G. It observes final outcomes rather than intermediate nodes. However, the computation of the utilities of the leaf nodes according to utility function U relies on the history of submissions of agents.

6.5.1.2. Simultaneous n-Player Game ($\mathbf{ARI_2}$)

The formalization of ARI_2 corresponds to the set-up of a simultaneous n-player game (section 3.2.3). The agent-rationality interface realizes a mapping

$$\begin{aligned} ARI_2 : G &\longmapsto SNP \\ \Leftrightarrow ARI_2 : (N, E, r) &\longmapsto (P, S, U). \end{aligned} \tag{6.4}$$

In this formulation N denotes nodes, E edges, and r the root of the state space graph G as described in section 6.4. These are mapped to a decision problem SNP with player set P, S as an m-tuple of pure strategy sets, one for each player, and F as an m-tuple of payoff functions (section 3.2.3).

Interpretation In the applied sense of the cooperative arrival management problem the components of SNP have the following meaning:

- The finite set P of players contains the four aircraft, labeled $1, 2, .., 4$.
- The strategy set S_k of each player $k \in P$ contains four strategies $S_k = \{1, 2, 3, 4\}$ denoting the four possible time points for submitting the ETA-shift allowed in the model.
- The m-tuple $\mathbf{S} = (S_1, S_2, S_3, S_4)$ contains the strategy sets of all four aircraft.
- The m-tuple $\mathbf{F} = (U_{1,node}, U_{2,node}, U_{3,node}, U_{4,node})$ contains the same function U according to equation 6.1 for each player.

Information availability/access The interface ARI_2 is based on very similar assumptions with regard to information availability as ARI_1. The major difference between ARI_1 and ARI_2 lies in the formulation as an n-player decision problem. For ARI_2, all n players act strategically and recognize the interdependency of their payoffs. The consequences of this assumption will be detailed further in the context of implemented rationality concepts and solvers (section 6.6).

6.5.1.3. Multistage n-Player Game ($\mathbf{ARI_3}$)

The formalization of ARI_3 corresponds to the set-up of a multistage game with n players. The agent rationality interface realizes a mapping

$$\begin{aligned} ARI_3 : G &\longmapsto MSG \\ \Leftrightarrow ARI_3 : (N, E, r) &\longmapsto (s, A, p, h). \end{aligned} \tag{6.5}$$

In this formulation $G = (N, E, r)$ is mapped to a multistage n-player decision problem MSG with a start s, a choice set function A, a probability assignment p and a payoff function h.

Interpretation In the applied sense of the cooperative arrival management problem the components of MSG have the following meaning:

- The start s of the game corresponds to the root node and representing the initial traffic situation.

- The function A is a player choice function, determining which aircraft can make which choices at which stage/point during the game.
- The probability p ist the probability of stochastic nature moves.
- The function h is a payoff function.

Information availability/access The ARI_3 interface makes a significantly different assumption regarding information-availability compared to ARI_1 and ARI_2. In contrast to the above formulations, the game here assumes that actors can in fact observe all past actions of other actors which have occured so far during the play. The game is played in several (here: four) stages. During each stage, actors can base their decisions on the history of interactions observed so far. Operationally this assumes aircraft which have knowledge about

- the resource allocation mechanism of the arrival manager,
- the payoff functions of the other agents,
- the traffic situation,
- the communication between other aircraft and the arrival manager, including which ETAs other aircraft have submitted to the arrival manager at what time.

Such a setting is practically credible mainly in a highly CDM-oriented environment where system-wide information management is in place (see SWIM, section 2.2). Extensive situation data are assumed to be available from a database to increase ATM actors' situational awareness. In practice, while sequence switches will always be directly observable, the availability of knowledge on all other actors' submissions is a much harder assumption.

However the assumption made here on information-availability has additional value as an extreme case. This can serve to test if such a setting would have negative effects with strategic, selfish agents.

If the system performance of ARI_3 in comparison to ARI_2 was reduced and it showed a significant increase in strategic interactions, this would mean that holding back information would actually protect the integrity of the system.

6.5.2. System-Optimality Interface (SOI)

The system-optimality interface (SOI) establishes measures of system performance from a global system point of view. It also provides access to agents' behavioral measures which are used for optimization. The implemented SOI realizes a mapping

$$\begin{aligned} SOI : G &\longmapsto OP \\ \Leftrightarrow SOI : (N, E, r) &\longmapsto (J, U, PT, Q, C). \end{aligned} \tag{6.6}$$

In the right-hand clause of the expression the elements are

- The set of joint utilities J, one for each leaf node n_{leaf}, computed as a sum of the individual utility values $U_{node,player}$ so that

$$J_{node} = \sum_{player=1}^{m} U_{node,player}. \tag{6.7}$$

- The set $\mathbf{U}$ with utility vectors U, one vector for each node, which contains as elements the individual utilities $U_{node,1}...U_{node,m}$ for players $1...m$ associated with that node (compare equation 6.1).
- The set of total process time PT, one for each node n_{leaf}. Total process time is defined as the sum of all individual aircraft flight times

$$PT_{node} = \sum_{player=1}^{m} TTA_{node,player}. \tag{6.8}$$

- The set of total quality values Q_{Total}, one for each node n_{leaf} (see equation A.12)
- The set of sequence changes, one for each node n_{leaf}. Each value C_i describes the total number of sequence changes C_{node} during the game history leading to that outcome (node). More formally, let P be the path from the root to node n_{leaf} consisting of the player nodes $p_o = r, p_2, p_3, p_4, p_5 = n_{leaf}$ on the path, then

$$C_{node} = \sum_{stage=2}^{5} \begin{cases} 1 & seq(p_{stage}) \neq seq(p_{stage-1}) \\ 0 & seq(p_{stage}) = seq(p_{stage-1}) \end{cases}. \tag{6.9}$$

6.6. Implemented Agent-Rationality Concepts

A rationality concept in terms of the DAVIC framework is a formal rule for determining what can be considered as rational behavior for an agent when interacting with the other agents via the planning system. Applied to the arrival management problem, this rule serves to determine the planning cycle when rational, self-interested aircraft should submit the modified ETAs to the arrival manager. It is assumed that aircraft make strategic choices regarding timing according to a certain principle in order to optimize their own sequence outcome.

Four different agent-rationality concepts have been implemented for the analysis of the arrival management application. All implemented solvers Sl_{AR} work on the same

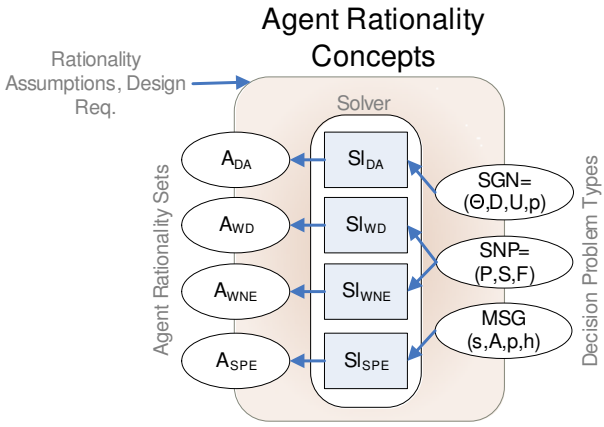

Figure 6.19.: DAVIC Agent-Rationality Concepts

state space. However, they use different problem formulations and therefore access the state space through the three different agent-rationality interfaces ARI_1, ARI_2 and ARI_3 (section 6.5.1). The outcome produced by each solver Sl_{AR} is a set of nodes A_{AR}. Each set A_{AR} represents all rational solutions according to the given rationality concept AR. Each node corresponds to one solution. Practically, in the presented example, a rational solution is a rational combination of aircraft choices of timing, according to a given concept.

The relation between decision problem types, solvers (to implement individual rationality concepts) and resulting agent rationality sets is illustrated in Figure 6.19. The four implemented solvers are introduced in the following. The decision-theoretic and game-theoretic foundations of all four solvers were discussed in section 3.2 from an application-independent perspective. The implemented solvers rely on a certain type of interface, but through the interface they are decoupled from their specific application and are therefore also reusable.

Decision Analysis Solver ($\mathbf{SL_{DA}}$)

The Decision Analysis (DA) solver Sl_{DA} relies on the agent-rationality interface ARI_1. This uses the problem formulation $SGN = (\ominus, D, U, p)$ (according to equation 6.3) of a simultaneous game against nature. The solution set AR_{DA} is achieved by maximization of expected utility EU. The decision analysis of this solver is done separately from the perspective of each individual agent. Meanwhile, from each agent's perspective, the other agents are assumed to make random choices, i.e. non-strategic choices 'by nature' (compare section 3.2.1).

From the application point-of-view, solver Sl_{DA} assumes an operational practice where all aircraft make efforts to maximize their expected utility. However, aircraft do not anticipate other actors' reasoning (either because they are unable to do so, or because they do not assume strategic behavior on the part of other aircraft).

No circular argument is conducted to base one's own decision on other aircrafts' reasoning.

Iterated Weak Dominance Solver ($\mathbf{SL_{WD}}$)

The Iterated Weak Dominance (WD) solver Sl_{WD} relies on the agent-rationality interface ARI_2. This solver employs the problem formulation $SNP = (P, S, F)$ (according to equation 6.4) of a simultaneous n-player game. The solution AR_{WD} is achieved by the iterated elimination of dominated strategies (IEDS) algorithm (see section 3.2.3).

The solver Sl_{WD} assumes an operational situation were true interactive decision-making of aircraft occurs. All aircraft not only strive to optimize their own results. They are also aware of the fact that other players do the same and the other players' reasoning is anticipated for their own decisions. IEDS is a simple but effective principle to determine solutions under such circumstances. The relative simplicity can make it practically more credible than more sophisticated rationality approaches. A potential weakness of the concept exists in the problem formulation of the application as a simultaneous game SNP, in which intermediate states are not observed. Also, IEDS results can depend on the order in which players are considered and elimination takes place (compare section 3.2.3).

Weak Nash Equilibrium Solver ($\mathbf{SL_{WNE}}$)

The Weak Nash Equilibrium (WNE) solver Sl_{WNE} relies on the agent-rationality interface ARI_2. In the same way as the Iterated Weak Dominance solver Sl_{WD}, it employs the problem formulation $SNP = (P, S, F)$ (equation 6.4) of a simultaneous n-player game. The solution set AR_{WNE} is determined directly according to the definition of a weak Nash Equilibrium. Algorithmically, the presence of a WNE is tested for each individual outcome (see section 3.2.3). Only pure Nash Equilibria are discovered, mixed strategies are not considered in the current implementation.

The solver Sl_{WNE} assumes that all aircraft strive to optimize their own results. Aircraft are also aware of the fact that other players do the same and anticipate their argument in the own decisions. An outcome is considered a game-theoretically stable solution (i.e. Nash Equilibrium) if no aircraft can profit by unilaterally deviating from that strategy.

The Nash Equilibrium concept is probably the most intensively studied and used in game theory. Again, the ignorance of intermediate states remains a limitation. Because of this, equilibria can result which would not be played by perfectly informed aircraft during the game. From a computational point-of-view, the WNE principle is more demanding than the IEDS. Related to this demand, the required decisions/ strategies are potentially less credible for an aircraft to perform in a real-world environment without dedicated technical system support.

Subgame Perfect Equilibrium Solver (SL_{SPE})

The Subgame Perfect Equilibrium (SPE) solver Sl_{SPE} builds on the agent-rationality interface ARI_3. In contrast to the above solvers, this is the only one that works on the problem formulation $MSG = (s, A, p, h)$ (according to equation 6.5) of a multistage n-player game. The solution set AR_{SPE} is determined directly according to the definition of a Subgame Perfect (Nash) Equilibrium (SPE). Again, the presence of SPE is tested for each individual outcome (see section 3.2.4).

The Sl_{SPE} solver is the only one of the four that assumes that aircraft can actually observe other participants' actions during each cycle of the application. Based on that information, aircraft make rational timing choices for ETA submission. They take into account that other aircraft apply the same reasoning and anticipate their argument in the own decisions. Note that by definition, the solution AR_{SPE} is a subset of AR_{WNE}. The former (AR_{SPE}) is also termed a refinement of the latter one (AR_{WNE}). The AR_{SPE} concept is the most sophisticated and computationally most demanding of the four. It is also the most difficult to apply for an aircraft in a real world environment.

6.7. Implemented System-Optimality Concepts

A system-optimality concept in terms of the DAVIC framework is a formal rule, stating how the game should be played from the designer's point of view. Applied to the arrival mangement problem, system-optimality concepts establish criteria regarding the desired behavior of aircraft for ETA submission from a global perspective. This desired behavior is necessary in order to realize the intended emergent system performance (i.e. throughput, flight efficiency, stability) of the overall arrival management system. It is also needed to keep the system workable for the human beings (i.e. air traffic controllers) involved.

Five different system-optimality concepts have been implemented for the analysis of the arrival management application, which are discussed in the following. The relation between basic system cost/utility functions provided by the SOI, derived system-optimality criteria and resulting system-optimality sets is illustrated in Figure 6.20.

Hicks-optimal set S_H The Hicks-optimal set S_H is computed on the basis of the joint-agent utility J (equation 6.7) provided by the SOI. The set S_H is defined as

$$S_H = \arg\max_{n \in N} J(n) \tag{6.10}$$

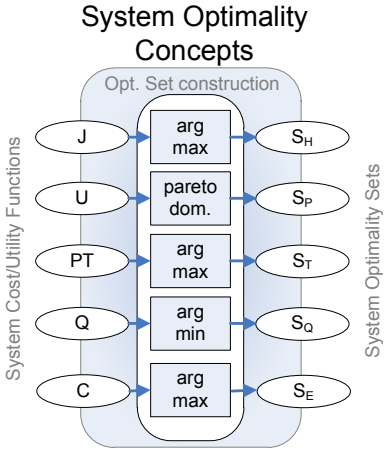

Figure 6.20.: DAVIC System-Optimality Concepts

where the argument J is the sum of individual agent utilities U. For the arrival-management problem U is defined as a weighted sum of flight time (negative sign, i.e. the lower the better) and reward for early submission (positive sign, i.e. the higher the better) (see equation 6.1). The Hicks-optimal set S_H thus contains solutions with optimal combinations of low overall flight time (which implies low fuel burn and high throughput) and low landing fees (due to no late changes and therefore high early-submission rewards) summed up over all aircraft.

Pareto-optimal set S_P The Pareto-optimal set S_P is the second measure of joint agent utility. It is computed on the basis of individual agent utilities U according to the principle

$$S_P \quad = \quad \{n \in N \mid \neg\exists\, n' \in N \; n' \succ n\}, \tag{6.11}$$

and thus includes all nodes which are not Pareto-dominated by other nodes (see definition 3.3.3).

Thus, S_P contains all outcomes of the arrival management process for which it is impossible for any aircraft to achieve better utility (lower flight time and lower landing fee (higher reward for early submission) without putting another aircraft off worse (higher flight time and/or higher landing fee).

Time-optimal set S_T The time-optimal set S_T contains all planning process outcomes with minimal process time PT. It is defined as

$$S_T = \arg\min_{n \in N} PT(n) \tag{6.12}$$

where PT is the sum of individual process times of all aircraft (equation 6.8). The time-optimal set S_T can be interpreted as a third measure of joint agent-utility. More importantly however, it is a measure of objective system performance. For any arrival management system- design, a central goal will be to maximize throughput of the system and minimize flight times of queuing aircraft.

Quality-optimal set S_Q The quality-optimal set S_Q contains all arrival process outcomes which realize a maximum of the quality value Q_{Total}. This quality value Q_{Total} is calculated by the arrival management mechanism during the process of sequence evaluation and selection to evaluate the best system response to a given input (section 6.3.2 and equation A.12). It is assumed that Q_{Total} represents the designer's view of what is a good quality sequence. Consequently, the behavior which produces good quality sequences are regarded as desirable. The criterion is defined as

$$S_Q = \arg\max_{n \in N} Q(n). \tag{6.13}$$

Early-move set S_E The Early-Move set S_E contains only a single outcome. It is the outcome that results when all aircraft submit their modified ETAs as soon as possible, i.e. in the first planning cycle. In DAVIC terminology, S_E is an agent behavioral measure. It describes a direct expectation of the kind of choices which should be made. The formulation of this expectation is in principle independent of the characteristics of the sequence produced by it (e.g. resulting performance measures such as flight time). The early-move set S_E is based on the value of C offered by the SOI (see equation 6.9) and is computed as

$$S_E = \arg\min_{n \in N} C(n). \tag{6.14}$$

Summary/discussion

Each of the system-optimality concepts S_H, S_P, S_T, S_Q, S_E is based on one of the five different basic measures J, U, PT, Q, C provided by the system-optimality interface SOI introduced in section 6.5.2. Among them are instances of joint-utility measures, agent- behavioral measures, and planning-system-quality measures.

The implemented system-optimality criteria formalize the system requirements established on the application layer. They provide different perspectives on the arrival management process and normative expectations of how the aircraft involved in it should act. For example, the perspectives of individual and aggregated flight times, landing fees, planning system quality measures and timing of ETA submission are directly analyzed in the different criteria. Note that expectations of different system optimality-criteria can turn out to be contradictory. Depending on scenarios and mechanism properties, it may not be possible to fulfill them simultaneously. In order to analyze the compatibility of the criteria, design-view intersections will be formulated in section 6.8.

6.8. Implemented Analysis Criteria

The criteria implemented for the analysis of arrival management mechanism build directly on the introduced agent-rationality sets and system-optimality sets. Specifically, they process as inputs

- the agent-rationality sets A_{DA}, A_{WD}, A_{WNE}, A_{SPE}, which describe *rational* aircraft behavior,
- the system-optimality sets S_H, S_P, S_T, S_Q, S_E which describe *desired* aircraft behavior.

As described in section 5.2.7 of the generic DAVIC framework, the analysis of these primary sets is structured in three steps, namely

1. node set integration (building derivative, secondary sets of the primary agent-rationality and system-optimality sets),
2. formal proofs of logical set relations (regarding rational and desired aircraft behavior),
3. quantitative evaluation of set sizes and properties.

The relation of all implemented analysis criteria for the arrival-management mechanism is illustrated in Figure 6.21. Each of the criteria introduced in the following is allocated to the three steps leading from primary sets to secondary sets and to the definition of formal and quantitative analysis criteria.

6.8.1. Primary and Secondary Sets

Design-view intersections

To find outcomes (if such exist) where aircraft comply with several of the design expectations at the same time, design-view intersections are established.

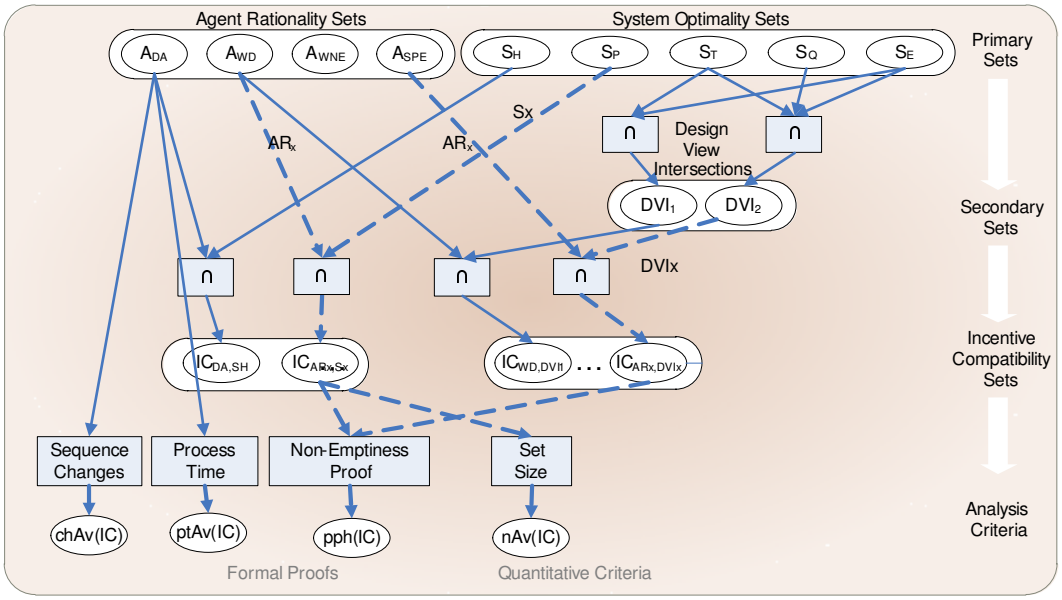

Figure 6.21.: DAVIC Analysis Layer

DVI_1 Design-view intersection DVI_1 combines the behavioral criterion of early submission of ETA modifications (S_E) with the criterion of time optimality (S_T). It contains outcomes where early ETA submission behavior also produces sequences with minimum process time. Formally, DVI_1 is defined as

$$DVI_1 = S_E \cap S_T \ . \tag{6.15}$$

DVI_2 Design-view intersection DVI_2 is a further refinement and subset of DVI_1. Specifically, DVI_2 contains all outcomes where aircraft submit early and produce a Hicks-optimal, time-optimal and quality-optimal sequence all at the same time. Design view intersection DVI_2 is formally defined as

$$DVI_2 \quad = \quad S_E \cap S_H \cap S_T \cap S_Q \ . \tag{6.16}$$

Incentive-compatibility sets

In order to find out which outcomes of the planning process satisfy certain agent-rationality and system-optimality criteria simultaneously, incentive-compatibility sets are established.

IC_{AR_x,S_x} In equation 6.17, an incentive-compatibility set $IC_{ARx,Sx}$ presents an intersection of one of the defined agent-rationality sets AR_x with one of the defined

system-optimality sets S_x, so that

$$\begin{aligned} IC_{ARx,Sx} &= A_{ARx} \cap S_x \\ AR_x &\in \{A_{DA}, A_{WD}, A_{WNE,}A_{SPE}\} \\ S_x &\in \{S_{H,}S_P, S_T, S_Q, S_E\}\ . \end{aligned} \tag{6.17}$$

The purpose of $IC_{ARx,Sx}$ is to define for which outcomes (if any) the participating aircraft act according to a rationality principle AR_x and at the same time fulfil the system-optimality criterion described by S_x.

$\mathbf{IC_{AR_x,DVI_x}}$ In equation 6.18 each of the incentive compatibility sets $IC_{ARx,DVIx}$ presents an intersection of one of the defined agent-rationality sets AR_x with one of the defined design-view intersections, so that

$$\begin{aligned} IC_{ARx,DVIx} &= A_{ARx} \cap DVI_x \\ AR_x &\in \{A_{DA,}A_{WD}, A_{WNE,}A_{SPE}\} \\ DVI_x &\in \{DVI_1, DVI_2\}\ . \end{aligned} \tag{6.18}$$

The purpose of $IC_{ARx,DVIx}$ is to define which strategies would be rational for participating aircraft and at the same time fulfill the combined expectations of the design-view intersection.

6.8.2. Formal Proofs

Formal proofs and quantitative analysis criteria are used within the DAVIC framework to analyze agent-rationality sets, system-optimality sets, design-view intersections and incentive-compatibility sets. To establish these criteria on the sets, the following definitions are introduced: Let DM_i be a set of outcomes resulting from the state space analysis of a single arrival-management scenario $Sc_i \in SC$. Let $pred$ be a predicate function defined on DM, so that $pred(dm) = true|false$ for any single node $dm \in DM$. Let T be the subset of DM with $T = \{dm : pred(dm) = true\}$. Let $\mathcal{T}$ be the collection of sets T_i for all individual Scenarios $Sc_i \in SC$. This means that $\mathcal{T} = \{T_1, T_2, ..., T_n\}$.

Percentage-proof-holds

The formal criterion $pph(\mathcal{T})$ (percentage-proof holds) is defined as the percentage of subsets $T_i \in \mathcal{T}$ for which T_i is not empty, as expressed by the equation

$$pph(\mathcal{T}) = \sum_{i=1}^{n} \frac{T_i \neq \emptyset}{n} \cdot 100\%\,. \tag{6.19}$$

This criterion *pph* relies on a formal proof of non-emptiness of T_i which is a result of the analysis of a single scenario. This proof of non-emptiness shows that e.g. a design expectation (represented by S_x) is compatible with rational behavior (represented by AR_x) for a given mechanism and scenario, as it proves that the intersection of these sets is non-empty. The individual non-emptiness proofs for T_i are aggregated to the percentage-proof-holds over the set collection $\mathcal{T}$.

6.8.3. Quantitative Analysis Criteria

Average set size

The average set size is an important quantitative measure to know the extension of agent-rationality sets, system-optimality sets and incentive-compatiblity sets and their sizes relative to the overall solution space $|DM|$.

Let $n(T)$ be the number of elements in T also written as $|T|$.

Then $nAv(\mathcal{T})$ is the average number of elements in the collection of sets $\mathcal{T}$, formally defined as

$$nAv(\mathcal{T}) = \sum_{i=1}^{n} \frac{n(T_i)}{n} \ . \tag{6.20}$$

Sequence changes

As discussed in the application layer section, the number of sequence changes is an important criterion for operability of a system. If the number of sequence changes is too high (the stability of the sequence not sufficient), operational inefficiencies will arise, human operators (i.e. air traffic controllers) will have difficulties in working the process and will have to spend too many resources in re-negotiations.

Let $ch(dm)$ be the number of sequence changes occurring on the system trajectory from the initial marking to the terminal node dm and let $chav(T)$ be the average number of sequence changes in the set of outcomes T.

Then $chAv(\mathcal{T})$ is the average number of sequence changes in the set collection $\mathcal{T}$, formally defined as

$$chAv(\mathcal{T}) = \sum_{i=1}^{n} \frac{chav(T_i)}{n} \ . \tag{6.21}$$

Process time

The overall process time is a central measure of system performance as discussed in the application layer section. Process time has to be controlled for all rational-

and system-optimality sets in order to compare the effects of different mechanism variants and achieve the desired emergent system performance.

Let pt be the accumulated process time of all aircraft for the terminal node $dm \in DM$ and let $ptav(T)$ be the average process time for a set of outcomes $T \in DM$.

Then $ptAv(\mathcal{T})$ is the average process time for the collection of sets $\mathcal{T}$ according to the equation

$$ptAv(\mathcal{T}) = \sum_{i=1}^{n} \frac{ptav(T_i)}{n} \ . \tag{6.22}$$

6.9. Definition of Design Space and Optimization Criterion

The central purpose of the DAVIC framework application is to conduct a systematic search over the design space DS, in order to identify the mechanism variant $M_{opt} \in DS$ which best satisfies (i.e. maximizes) the quality function C_{ov}. This section defines the optimization criterion C_{ov} and the design space DS of sequencing mechanisms M.

Optimization criterion

The criterion C_{ov} for measuring the quality of the individual mechanism variant M is motivated by the operational requirements discussed on the application layer (sections 6.1 and 6.2). It combines several analysis criteria introduced in section 6.8 evaluating the three aspects of

1. incentive-compatibility,
2. sequence stability, and
3. process time.

Incentive-compatiblity For this aspect, it is analyzed for which percentage of scenarios Sc rational agent-behavior is compatible with the defined system-optimal behavior, for a mechanism M. The analysis relies on the definition of the percentage-proof-holds pph, see equation 6.19.

Let ic_s represent incentive-compatibility of the agent-rationality set A_x with the system-optimality sets S_H, S_P, S_T, S_Q, and S_E

$$\begin{aligned} ic_s \ = \ & pph(\boldsymbol{IC_{A_x,S_H}}) + pph(\boldsymbol{IC_{A_x,S_P}}) + pph(\boldsymbol{IC_{A_x,S_T}}) + \\ & pph(\boldsymbol{IC_{A_x,S_Q}}) + pph(\boldsymbol{IC_{A_x,S_E}}) \ . \end{aligned} \tag{6.23}$$

Further, let ic_d represent incentive-compatibility of $A_x$with the design view intersections

$$ic_d = pph(\boldsymbol{IC_{A_x,S_{DVI_1}}}) + pph(\boldsymbol{IC_{A_x,S_{DVI_2}}}) \tag{6.24}$$

Then the value ic_{ov} represents all formal incentive-compatibility measures such that

$$ic_{ov} \quad = \quad ic_s + ic_d \ . \tag{6.25}$$

Sequence stability For the aspect of sequence stability, the average number of sequence changes $chAv$ in the case of rational agent behavior AR_x is evaluated by computing $chAv(\boldsymbol{AR_x})$as formerly defined in equation 6.21.

Process time The process time for mechanism variant M in case of rational behavior is compared to the minimum possible process time.

Let

$$ptAv_{Dif} \quad = \quad ptAv(\boldsymbol{AR_x}) - ptAv_{Min}(DS) \tag{6.26}$$

with

$$ptAv_{Min} \quad = \quad \arg\min_{M \in DS} ptAv(M). \tag{6.27}$$

That is, $ptAv(\boldsymbol{AR_x})$ is the average process time (across all scenarios) for rational behavior with the mechanism variant M_i. Further $ptAv_{Min}(DS)$ is the fastest average process time (across all scenarios) for any of the mechanism variants $M \in DS$. That is, $ptAv_{Dif}$ evaluates the extra process time needed with M_i in comparison to the fastest mechanism $M \in DS$.

Overall mechanism quality Finally, the overall optimization function C_{ov} is defined as the weighted sum of the three components, ic_s for incentive-compatiblity, $chAv$ for number of sequence changes, and $ptAv_{Dif}$ for excess process time

$$C_{ov} = w1 \cdot ic_{ov} + w2 \cdot chAv + w3 \cdot ptAv_{Dif} \ . \tag{6.28}$$

The weights w_1,w_2, and w_3 define the priorities and trade-offs between incentive-compatibility, sequence stability and process time. Parameters $w2$ and $w3$ will have negative values as sequence changes and extra process time shall be minimized.

Design Space

The Design Space DS describes the set of mechanism variants M which are considered for the selection of the optimal mechanism M_{opt}. For the arrival management example, the design space is spanned by the three dimensions of

1. type of stability function $SFtype$,
2. stability ratio sr,
3. reward for early submission esr.

Type of stability function In the first dimension of DS, one of the three alternative stability functions SF_1,SF_2,SF_3 is selected, which were introduced in section 6.3.2. That is

$$SF_x \in SFtypes = \{SF_1, SF_2, SF_3\} \ . \tag{6.29}$$

Stability ratio In the second dimension of DS, the weighting of the selected stability function SF_x with regard to the optimization function $Q_{EarlyETA}$ is expressed by the stability ratio sr . The parameter sr defines the relative contribution of the two functions towards Q_{Total} (compare equation A.12) so that formally

$$sr = \frac{w_{SF}}{w_e} \cdot 100\% \ . \tag{6.30}$$

For the analysis in chapter 7 the set of discrete values for sr will be

$$sr \in SR = \{0\%, 1\%, 2\%, \ldots, 20\%\} \ . \tag{6.31}$$

Reward for early submission In the third dimension of DS, a dedicated monetary reward to agents who submit their ETA early can be either activated or deactivated. The parameter

$$esr \in ESR = \{0, 1\} \tag{6.32}$$

determines if this reward is active ($esr = 1$) or not ($esr = 0$). As explained in the agent-utility definition (equation 6.1), the size of the reward itself is a function of timing. The earlier the planning cycle in which the change is submitted, the higher the reward.

Design space The design space DS considered for the analysis with DAVIC is finally defined by the Cartesian product for the three dimensions

$$DS = SFtypes \times SR \times ESR \ . \tag{6.33}$$

The result is graphically shown in Figure 6.22. With the assumed discretization, DS contains $3 \cdot 21 \cdot 2 = 126$ mechanism variants M_i.

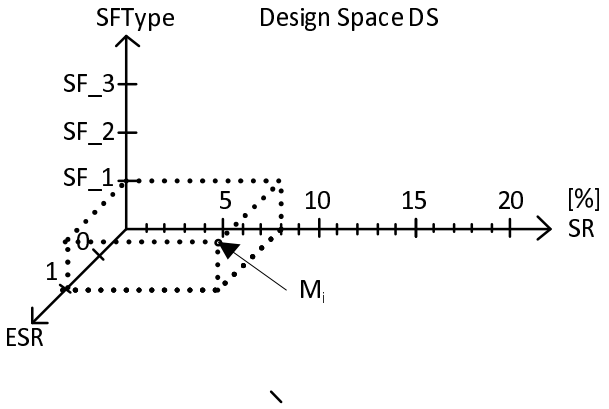

Figure 6.22.: DAVIC - design space definition with dimensions $SFtype$ (stability function), SR (stability ratio) and ESR (reward for early submission).

7. Results for Incentive-Compatibility and Optimization of System Performance

This chapter presents the results of the application of the DAVIC framework to Cooperative Arrival Management. It reports and discusses the outcomes of the specific analysis set-up developed in chapter 6. This chapter thereby demonstrates the use of the DAVIC framework for the analysis of incentive-compatiblity and optimization of emergent system performance on a specific mechanism design problem.

In a very abbreviated form, the two most important inputs to the analysis are restated:

- The *discrete design space* DS of potential sequence-planning mechanism M_i, spanned by variation of a) stability function type, b) stability ratio, and c) early-submission reward, and
- The *optimization criterion* C_{ov} (see equation 6.28) for the selection of the optimal mechanism variant M_{opt} for a given weighting of partial criteria a) incentive-compatibility, b) sequence stability, and c) process time.

As described in the set-up of the analysis in chapter 6, the optimization criterion C_{ov} is a complex, multi-layered criterion building on more basic elements. This chapter first presents the results for the basic elements and then, to ensure their correct interpretation, the aggregated results. The presentation follows the same logic in which the criteria are built on each other in chapter 6. The analysis begins with the following five steps:

S1) Analysis of the ability of the implemented agent-rationality solvers to predict rational choices for the mechanisms in DS. This leads to the selection of the most appropriate solver Sl (section 7.1)

S2) Assessment of the size of system-optimality sets S_x (section 7.2)

S3) Assessment of the inner consistency of design-requirements (expressed by design-view intersections) as a function of the mechanism variant M_i (section 7.3)

S4) Assessment of incentive-compatibility by contrasting *rational* behavior with *system-optimal* behavior (section 7.4)

S5) Analysis of the expected number of sequence changes and the expected process time under the assumption of rational agent behavior as well as system-optimal behavior (section 7.5)

For each of these five steps, the result is a function of the mechanism variant $M_i \in DS$. That is, the influence of the characteristics of M_i on the result is studied in terms of the three design-space dimensions (stability function, stability ratio, and reward for early submission).

In the following step, the direction of the analysis is reversed, to solve the central mechanism-design problem by

S6) Selection of the of optimal mechanism variant. This step serves to identify a mechanism variant M_{opt} (defined by the parameters SF_{opt},sr_{opt}, and esr_{opt}) in design space DS, so that C_{ov} is maximized (section 7.6).

Finally, the emergent system behavior which M_{opt} is expected to produce is looked at in the last step

S7) Effects on incentive-compatibility, process time and sequence stability (section 7.7).

The chapter is summarized in subsection 7.8.

7.1. Analysis of Agent-Rationality Sets

This subsection presents results of the non-emptiness proof and the size of agent-rationality sets A_x for the different AR-concepts Decision Analysis (DA), Weak Dominance (WD), Weak Nash Equilibrium (WNE) and Subgame Perfect Equilibrium (SPE) as a function of the different mechanism variants M in the design space DS. The results are important to judge the eligibility and usefulness of the four rationality concepts for the Cooperative Arrival Management application. They are used to select a solver S_l for the subsequent optimization tasks.

7.1.1. Non-Emptiness proof of Agent-Rationality Sets

The graphs in Figure 7.1 show the results for the non-emptiness proof $pph(A_x)$ of agent- rationality sets. That is, the percentage of games is calculated in which the set of agent-rational nodes procuced by a solver Sl_x is non-empty.

Each of the subfigures 7.1 left) and right) contains results $pph(A_x)$ for the four different A_x (agent-rationality) sets DA (Decision Analysis), WD (Weak Dominance), WNE (Weak Nash Equilibrium) and SPE (Subgame Perfect Nash Equilibrium). In both subfigures, the percentage $pph(A_x)$ of scenarios/cases where the non-emptiness proof of an AR-set holds is plotted against the stability-ratio (sr) dimension of DS over the range of $sr = [0\%, 20\%]$. The left subfigure presents results for an esr (early-submission reward, see section 6.5.1) setting of $esr = 0$ (i.e. deactivated), the right subfigure shows results for an esr setting of $esr = 1$ (i.e. activated).

The results in this Figure are averaged over the different stability functions SF_1, SF_2, SF_3. Potential differences in $pph(A_x)$ between individual stability functions for the selection process are not considered at this stage.

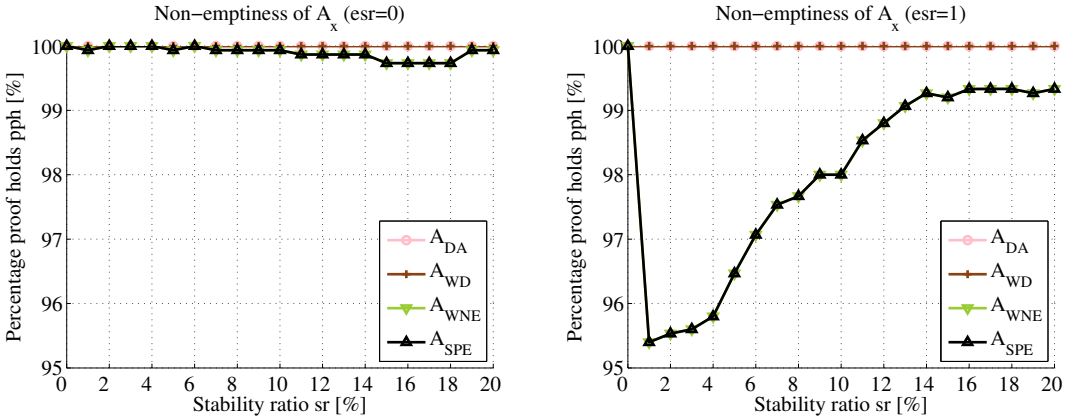

Figure 7.1.: Non-emptiness proof for rational sets as percentage $pph(A_x)$ - analyzing the percentage of scenarios where the individual agent-rationality solvers are able to make predictions

Results

- *Solvers Sl_{DA} and Sl_{WD}:* The solvers Sl_{DA} and Sl_{WD} are able to make predictions for agent- rational sets in all cases ($pph(A_x)$=100%), regardless of the specific parameter settings for *sr* and *esr*. This is in accordance with the theoretical expectations (subsections 3.2.1, 3.2.3).
- *Solvers Sl_{WNE} and Sl_{SPE}* : The solvers Sl_{WNE} and Sl_{SPE} (implementing the Nash Equilibrium concepts) produce a prediction for the majority (i.e. greater 95 %) but not all cases. Up to 5% of proofs fail in the right subfigure for the setting of $esr = 1$ and up to 0.5 % in the left subfigure for $esr = 0$. The reason is that no pure but only mixed NEs exist for some of the games, and mixed NEs are not currently computed by the solver. This circumstance is detailed below in the paragraph 'Interpretation of empty agent-rationality set'.
  Values for *pph* depend on the stability-ratio and the early-submission reward *esr*-setting (compare upper Figures left) and right)). For both WNE and SPE, the percentage where the non-emptiness proof holds is greater at the borders of the *sr*-range (i.e. for $sr = 0\%$ and $sr = 20\%$) than in between. Further, it can be observed that for WNE and SPE concepts, the plots are identical over the whole range of *sr* in both graphs. That means that in the cases where the game possesses a WNE, it possesses also an SPE.

Interpretation of empty agent-rationality set

The reason why the solvers Sl_{WNE} and Sl_{SPE} produce empty agent-rational sets A_x for some of the scenarios and mechanism variants is that they are constructed

to compute only pure Nash Equilibria (definition 3.2.9) and not mixed equilibria (definition 3.2.13). While it was stated in section 3.2.3 that all games have at least one Nash Equilibrium when allowing mixed strategies (definition 3.2.14), not all games possess a pure Nash Equilibrium. And in fact, some of the games realized by the combination of mechanisms and scenario instances Sc_i in this investigation do not possess pure Nash Equilibria. This results in an empty agent-rationality set A produced by the solvers Sl_{WNE} and Sl_{SPE}.

State space example for empty agent-rationality set

The general argument is illustrated for a specific example on the partial game tree in Figure 7.2. A fraction of a game tree is represented which has been derived by state space calculation of the Coloured Petri Net model for the following setting: stability function SF_3 with $sr = 30$, and $esr = 1$ and scenario instance Sc_{24}. On the left side of the Figure, the root node N4 is visible. Different node paths are shown leading to a number of terminal nodes N1219, N1216, N1217 etc. Each terminal node represents a different strategy profile and potentially different sequence- and utility outcomes.

During the analysis, the solver Sl_{WD} has identified the terminal nodes N1219, N1216, N1169 and N1176 as the agent-rationality set A_{WD}. For many games in SC this is identical with the set A_{WNE}. In these cases, however, none of these terminal nodes (and no other outcome of this game either, as could be shown) presents a Nash Equilibrium NE. For every single one of these outcomes, a non-empty list of other outcomes can be named which represent unilateral and profitable deviations of individual players from this state.

In the Figure, the node numbers identifying these profitable deviations are listed in the field NC (Nash counterexamples) of each terminal node. Illustrating the fact on an individual node, node N1219 does not represent a Nash Equilibrium NE since N1216 and N1217 represent profitable unilateral deviations of player D from outcome N1219. While all other agents make identical choices, if D delays the submission of the modified ETA ($\triangle eta_D$=44), it achieves a modified sequence of ACDB instead of ACBD and improves its utility to $u_D = -209$ or $u_D = -210$ instead of $u_D = -282$ for Node 1219. A similar argument could be made for all other terminal nodes in the game tree. The argument for the WNE solver extends to the SPE solver in a similar manner. As the game does not contain pure NE, it cannot contain any refined subgame perfect NEs either.

On the basis of the example, it becomes possible to explain in more detail the curve characteristics for WNE and SPE in Figure 7.1. For the early-submission reward setting of $esr = 1$, the lowest number of proofs hold for the case with the lowest non-zero stability ratio, while $p = 100\%$ for $sr = 0$ and monotonously increasing from $sr = 1$ upwards. Analytically, when esr is zero, any order of submissions and combination of agents' decisions leads to the same outcome, thus all outcomes are NEs. When sr is non-zero and increasing, an increasing number of games become completely stable and all sequence outcomes are identical with the initial situation. In this case all outcomes of this game become NEs, too.

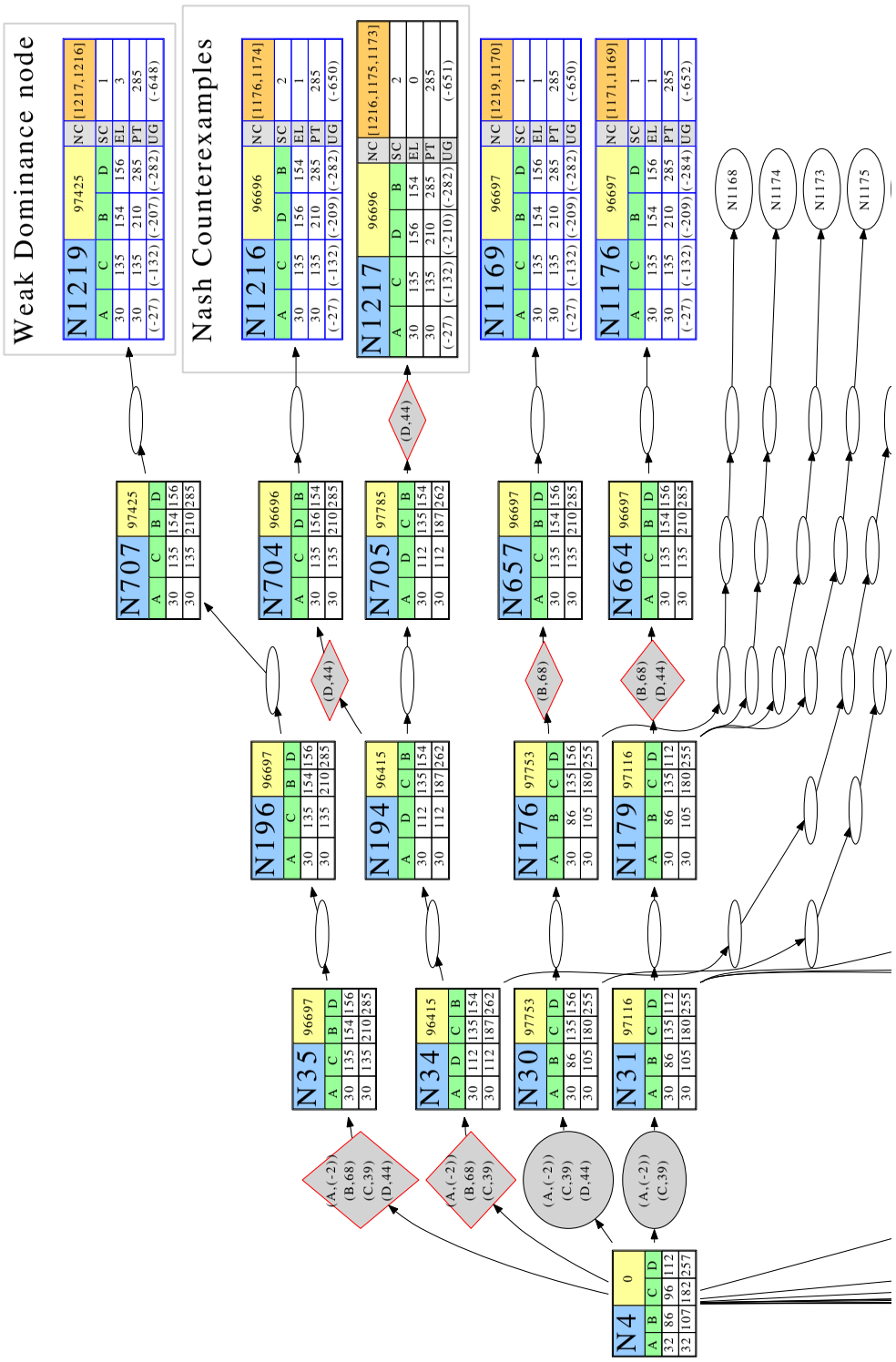

Figure 7.2.: Example case for empty sets A_{WNE} and A_{SPE} - state space representation for absence of pure Nash Equilibria and examples of profitable deviations, i.e. 'Nash Counterexamples' (Mechanism variant: SF_3, $sr = 3\%$, $esr = 1$, Scenario: Sc_{24})

7.1.2. Set Size of Agent-Rationality Sets

The graphs in Figure 7.3 show the average size nAv (equation 6.20) of the agent-rationality sets represented by the mean number of nodes in the set A, averaged over the set of scenario specifications SC. Results are represented by one solid line for each solution concept. The range of minimum and maximum sets sizes over all scenarios Sc_i is also indicated by shaded surfaces in the respective colors. As in Figure 7.1, results are plotted against the stability ratio sr. The left subfigure is based on a reward for early submission $esr = 0$ and the right subfigure on a setting of $esr = 1$. Note that the axis scaling of the ordinate is different between the left and right subfigure.

Maximum set size

With the degrees of freedom (DOF) of the four aircraft A, B, C, D in the simulation given, the maximum total number of the set of all end states Z_i for an individual scenario is 256 nodes (i.e. $options^{player} = 4^4 = 256$, see section 6.4). For some scenarios, however, one or more agents may have an ETA-shift $\triangle eta = 0$. This factually eliminates the options of these actors and reduces the potential number of outcomes in the decision tree. The factual average set-size of Z for the specific scenario set SC is 250.24 nodes. Thus an agent-rationality set-size nAv of 250.24 nodes would mean that all possible outcomes in all terminal node sets Z_i are members of the respective agent rationality set.

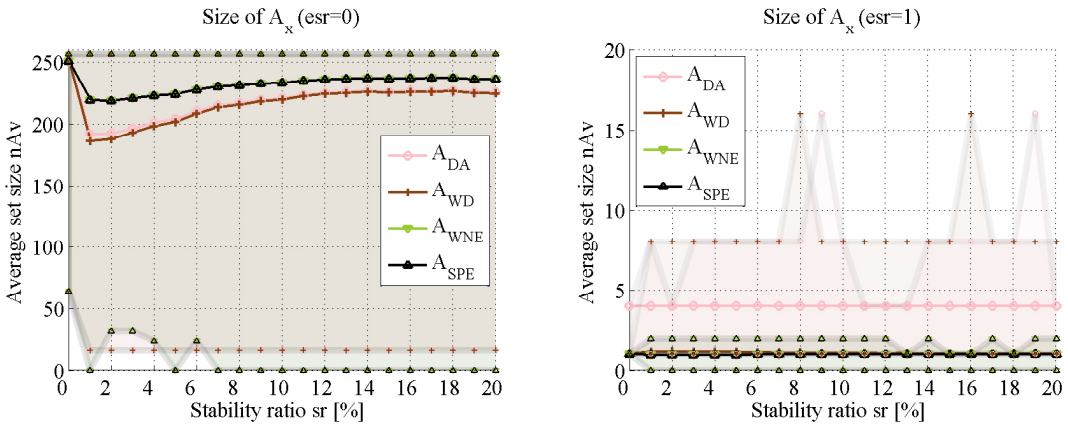

Figure 7.3.: Average, minimum, and maximum size of rational sets A_x predicted by individual solvers

Results

- Early-submission reward *inactive* (setting $esr = 0$, Figure 7.3 left):
  - The agent-rationality sets A_x cover the great majority of potential end states in Z. Averaged over all scenarios/games, between 80-90% of the

nodes in Z_i constitute rational plays depending on the given rationality concept. Notable differences exist mainly between the average set-sizes for A_{DA} and A_{WD} on the one hand and A_{WNE} and A_{SPE} on the other hand.

- For all solvers the maximum sizes of A reach the theoretical maximum of 256 nodes. The minimum sizes go down to below 50 nodes and can turn zero (empty rational set case, see above) depending on the rationality concept and the *sr* setting.

- Early-submission reward *active* (setting $esr = 1$, Figure 7.3 right):

  - Although the absolute cost/utility impact of $esr = 1$ is small, average rational set-sizes become very limited. In most cases A_x consists only of a single node for the A_{WD},A_{WNE} and A_{SPE} cases and always of four nodes for the A_{DA} case. The introduction/activation of the reward for early submission leads to the fact that now all nodes which represent delayed submission of information by an aircraft are inferior to the early-submission nodes. Delay of information does therefore not constitute rational behavior unless it effectively leads to a position-gain for that aircraft.

7.1.3. Discussion

Selection of agent-rationality concept

The aim of section 7.1 has been to investigate the ability of the four different agent-rationality (AR) concepts to predict and explain agent behavior for the application. Besides the ability to produce a solution at all, the size of the predicted sets allows some conclusions regarding the selectivity and sharpness of the prediction. Together with theoretical considerations on the underlying rationality concept, these data guides the selection of the most appropriate solution concept for the remaining analysis.

Of the four considered agent-rationality concepts, the weak dominance (WD) concept stands out positively for a number of reasons:

- The WD concept is able to predict a rational play for all scenarios. That is for both $esr = 0$ and $esr = 1$ it results $pph(A_{WD}) = 100\%$.
- Its theoretical assumptions are weaker than those of the WNE and SPE concepts. These assumptions are potentially more plausible for the description of real agents' behavior under limited time availability and limited cognitive resources (see section 3.2).
- In contrast to the DA concept, the WD concept reflects true interactive decision-making in the game-theoretic sense with interdependent payoffs and multiple

rational agents. Decision Analysis DA just models the game of one rational agent against nature.

In the context of this work, the arguments listed above lead to the selection of the weak dominance (WD) concept as the preferable rationality concept for the further analysis. In the following subsections, whenever a statement regarding rational choices is needed, the behavioral predictions of A_{WD} will used.

Effect of early-submission reward

A substantial effect of esr on the size and non-emptiness of rationality sets has been found. Introducing a reward for early submission of ETA-shifts $\triangle eta$ substantially reduces the size of rationality sets A_x (including A_{WD}). Rationality sets then contain nodes (strategy profiles) with late submission only where they lead to a direct benefit in form of a time-gain for at least one of the participating agents.

7.2. Analysis of System-Optimal Sets

This subsection discusses the size of the different system-optimal sets S_x as a function of the type of stabiliy function SF, the stability-ratio sr and the early-submission reward esr. As indicated in chapter 6 with the introduction of system-optimality sets, the non-emptiness of these sets is guaranteed by their definition. Thus, non-emptiness does not need not be further proven specifically for this application. However, the results for the sizes of the primary system-optimal sets S_x are important for the correct interpretation of the secondary sets derived from them in sections 7.3 and 7.4.

Four system-optimal sets are considered in the following:

- Hicks-optimal set S_H (equation 6.10)
- Pareto-optimal set S_P (equation 6.11)
- Time-optimal set S_T (equation 6.12)
- Quality-optimal set S_Q (equation 6.14)

Analysis approach

The plots in the following sections present results on the average, minimum, and maximum number of nodes of the respective optimality set S_x plotted against the stability ratio sr. Within each plot, three individual results for the different stability functions SF_1, SF_2, SF_3 are opposed. Results for averages nAv are indicated by solid lines in their respective colors (SF_1=blue, SF_2=green, SF_3=red) for each of the stability functions. Minimum and maximum values are represented by shaded surfaces in the same color. In Figures where nAv is a function of the early-submission reward esr, results for $esr = 0$ and $esr = 1$ are opposed in left and right subfigures.

Set size vs. inner set characteristics

Note that the same average set size nAv (and the same minimum and maximum sizes respectively) do not implicate that the underlying sets have the same characteristic according to any given quality measure. It merely states that the same number of nodes belongs to this set. Thus, $nAv(sr_1) = nAv(sr_2)$ does not implicate that some quality measure on the two set-collections will also lead to the same result, even if a quality measure (e.g. minimum total process time) defines the membership of the set. This statement applies to S_H, S_P, S_T, and S_Q. Results for quantitative quality measures as a function and characteristic of the different sets S_x are discussed below in subsection 7.5.

7.2.1. Hicks-Optimal Set

The Hicks-optimal set S_H denotes a subset of nodes in Z with the maximum joint utility for all agents as defined by equation 6.10.

Results

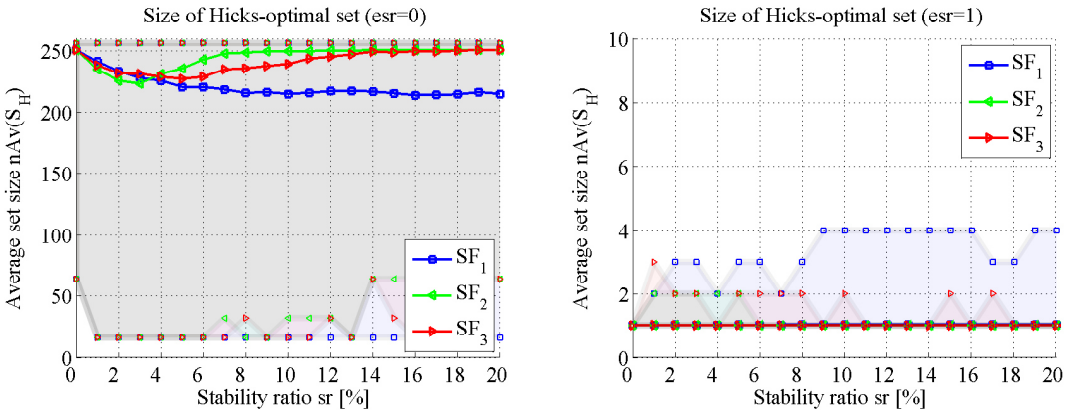

Figure 7.4.: Average size nAv of Hicks-optimal set S_H

The results for $nAv(S_H)$ are shown in Figure 7.4.

- Early-submission reward *inactive* (setting $esr = 0$, Figure 7.4 left):
  - For SF_2 and SF_3 the case of $esr = 0$, 100% of the nodes in Z belong to the Hicks-optimal set S_H for the lower and upper ends of the stability range. This is explained as follows: For $sr = 0\%$, all outcomes in Z are identical and correspond to the ETA-optimal sequence seq_{eopt} (i.e. the sequence is in the order of the ETAs). For $sr = 20\%$, all sequence outcomes within Z are identical and correspond to the initial sequence Seq_{ini}. As all sequence outcomes are identical, it follows that they all represent the maximum total utility, too, regardless of the absolute value

of this utility. For the intermediate range of sr, however (eg. $sr = 3\%$), a reduced size of the Hicks-optimal set S_H can be observed. In these cases, the stability function introduces time-dependent outcomes to the system, thus some games now allow for different sequence-outcomes. Some of the strategies available to agents in these cases cease to yield optimal solutions. However, as can be observed, the average size nAv of S_H remains above 80% of the potential maximum of 250.24 nodes. Thus, the great majority of nodes still belongs to the Hicks-optimal set S_H.

  - Unlike SF_2 and SF_3, stability function SF_1 does not lead to complete stability at the upper end of the stability scale with $sr = 20\%$. In fact, it does not realize complete stability for any potential value of sr. Thus, the outcomes continue to depend on the order of the submission of the ETA-shifts. Some of these outcomes are not Hicks optimal. This leads to values of around 80% of the maximum size at the upper end of the sr-range.

- Early-submission reward *active* (setting $esr = 1$, Figure 7.4 right)*:*

  - For the case of $esr = 1\%$, most games yield exactly one Hicks-optimal node in Z. This is the node which combines the lowest sum of aircraft flight times (total processing time pt) with the earliest possible submission leading to that sequence outcome. For some scenarios, however, up to four Hicks-optimal nodes can be found. An exemplary game where this is the case is discussed in the following.

Tree-based representation - multiple Hicks-optimal outcomes

An example case for the occurrence of multiple Hicks-optimal solutions is given in Figure 7.5, showing how multiple different solutions which all feature the maximum joint payoff for all agents can exist within one game. The presented game is defined by scenario Sc_{44} with a stability-ratio of $sr = 3\%$, early-submission reward $esr = 1$, and stability function SF_2. Two Hicks-optimal solutions exist, represented by the outcomes N1028 and N1068. Note that these two nodes share the same utility J (i.e. J=-436, in field 'UG' in the node label) which is the maximum joint utility for the game. Both solutions, which represent the same sequence outcome BCAD, are caused by different strategy profiles SP, however. They are also associated with a different number of sequence changes (ch(N1028)=1, ch(N1068)=2). While N1068 represents a Nash Equilibrium NE, N1028 does not.

7.2.2. Pareto-Optimal Set

The Pareto-optimal set S_P denotes the subset of outcomes within Z where, as defined by equation 6.11, it is not possible to make one agent better-off without making another agent off worse.

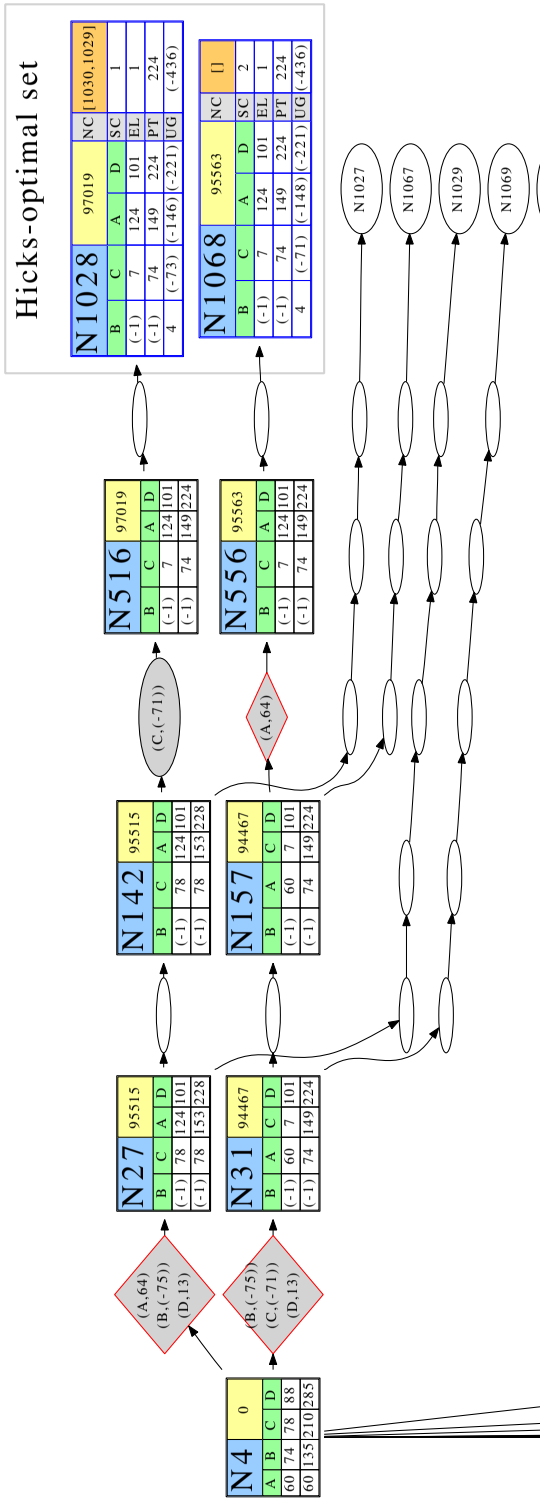

Figure 7.5.: Fraction of a game with multiple Hicks-optimal outcomes in state space representation (mechanism variant: SF_2, $sr = 3\%$, $esr = 1$, scenario: Sc_{44})

Results

The results for $nAv(S_P)$ are shown in Figure 7.6.

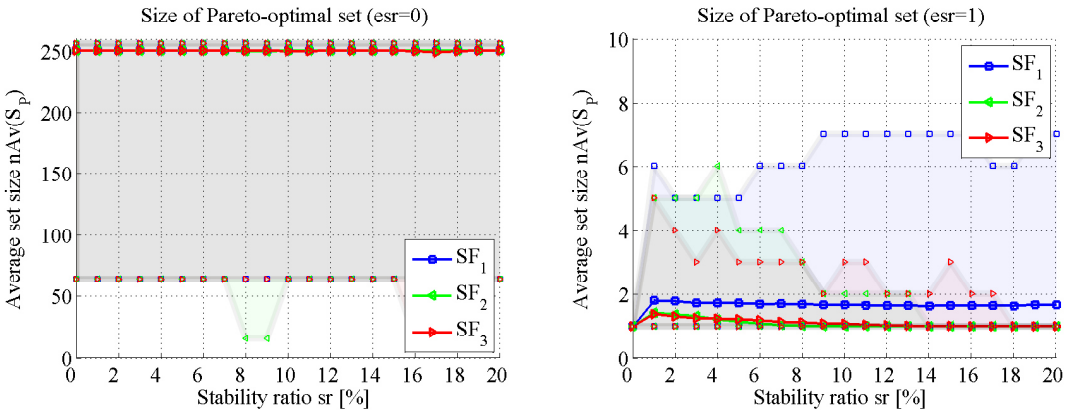

Figure 7.6.: Average size nAv of Pareto-optimal set S_P

- Early-submission reward *inactive* (setting $esr = 0$, Figure 7.6 left) :
  - For the case of $esr = 0$, 100% of the nodes in Z are Pareto-optimal for all settings of the system and all stability functions. Analytically, as there is no cost for late submission, the only factor determining the payoff is the resulting sequence position and the related TTA. Since it is not possible for an agent to improve his own sequence position and TTA without another agent suffering a position loss and delay, all outcomes in Z are Pareto-optimal outcomes.
- Early-submission reward *active* (setting $esr = 1$, Figure 7.6 right):
  - For the case of $esr = 1$, the average set size $nAv(S_P)$ is between 1 and 2 nodes, the maximimum set size of S_P is 7 nodes. As in this case, there is a reward for early submission (or a cost for late submission interchangeably), a potential sequence outcome usually corresponds to one Pareto-optimal outcome. The Pareto-optimal outcome is the agent strategy-profile with the earliest submissions,and therefore greatest early-submission reward, which leads to this sequence. In rare cases, different strategy profiles may yield the same sequence outcome and also have the same early-submission reward, then producing more than one Pareto-optimal node per sequence outcome. Such an example is demonstrated in the following.

Tree-based representation - multiple Pareto-optimal outcomes

In Figure 7.7 an example case from a game with three Pareto-optimal nodes is shown. The game was defined from scenario Sc_{10}, with the mechanism variant M_i with $sr = 3\%$, $esr = 1$, and stability function SF_1. The three Pareto-optimal nodes

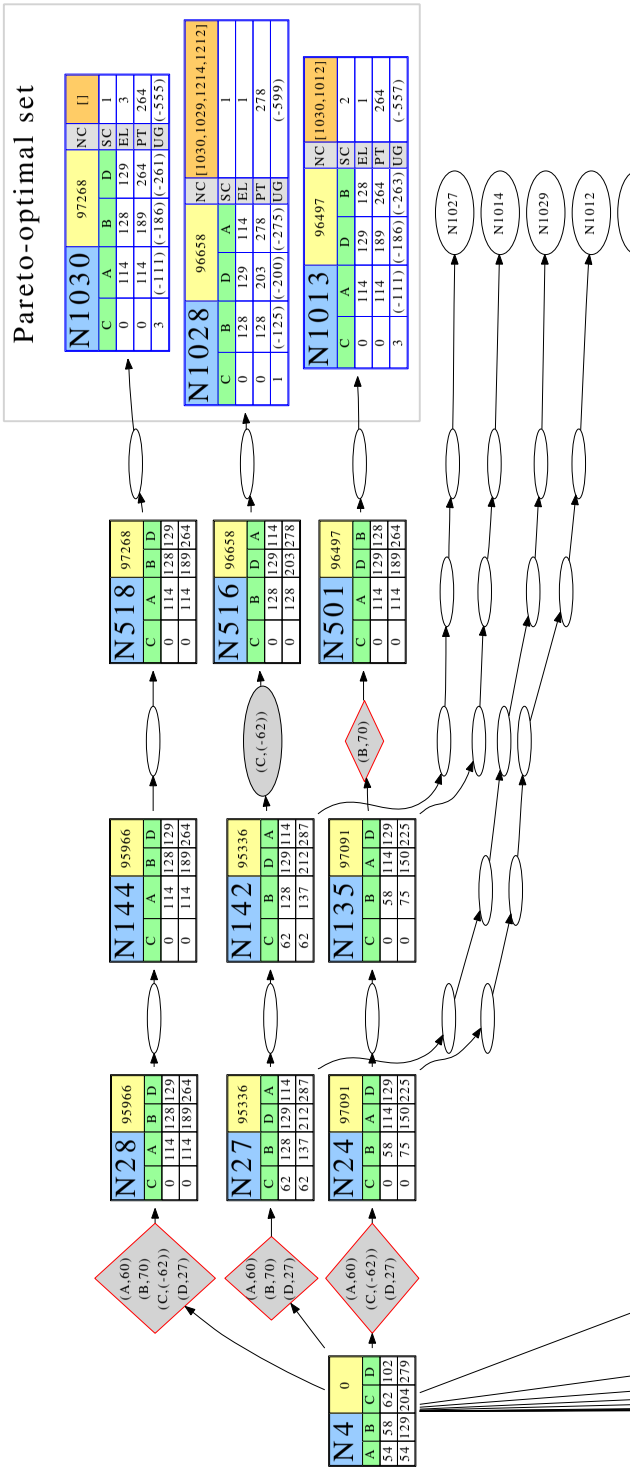

Figure 7.7.: Fraction of a game with multiple Pareto-optimal outcomes (mechanism variant: SF_1, $sr = 3\%$, $esr = 1$, scenario: Sc_{10})

N1030, N1028, N1013 represent three distinct sequence outcomes (CABD, CBDA, CADB). They are also associated with three different utility values J. However, the three nodes share the property that it is not possible to change to another node and improve one agent's utility without making at least one of the other agents off worse. Note that N1030 does also represent a Nash Equilibrium NE. The other two nodes (N1018, N1013) are no NEs.

7.2.3. Time-Optimal Set

The time-optimal set S_T denotes the subset of nodes within the terminal nodes Z with the minimum total process time pt, as defined in equation 6.12. The result for S_T is independent of the activation ($esr = 1$) or deactivation ($esr = 0$) of the early-submission reward.

Results

The results for $nAv(S_T)$ are shown in Figure 7.8.

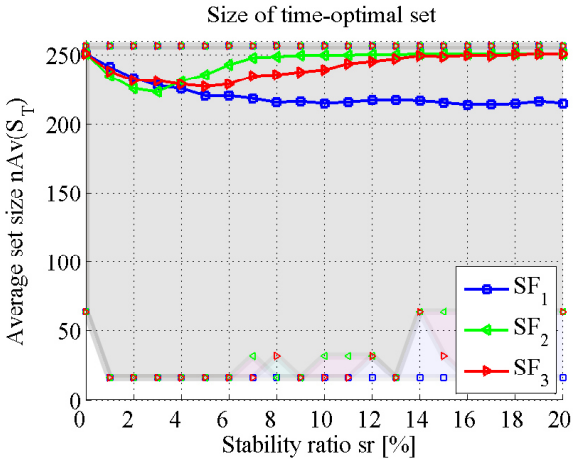

Figure 7.8.: Average size nAv of time-optimal set S_T

The time-optimal set S_T is mathematically identical with the Hicks-optimal set S_H for the setting of $esr = 0$ (Compare equation 6.7 and equation 6.8). Consequently, a complete correlation is observed between the results presented in Figure 7.8 and Figure 7.4 left. The same comments on results apply (compare section 7.2.1).

7.2.4. Quality-Optimal Set

The quality-optimal set S_Q set denotes the subset of nodes within Z which realize the maximum quality value $max(Q_{tot}(Z))$ in terms of the planning-mechanism's quality function Q_{tot}. Set S_Q was formally defined in equation 6.13. It is independent of the

setting of the parameter *esr*. The quality-optimal set is assumed to represent the system designer's will, which is implemented in the planning system.

Results

The results for $nAv(S_Q)$ are shown in Figure 7.9.

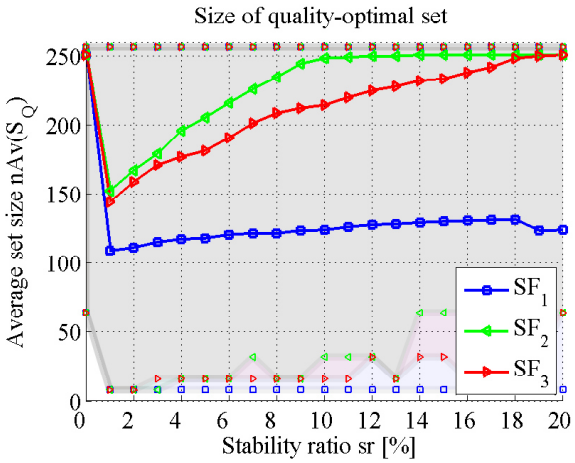

Figure 7.9.: Average size nAv of quality-optimal set S_Q

- *Effects of stability ratio:* The size of S_Q is 250.24 for SF_2 and SF_3 at the lower and upper ends of the range of sr, where therefore S_Q covers 100% of Z. However, a sharp decrease in set size can be observed for the lowest non-zero weightings of stability functions, with a minimum at an sr- value of $sr = 1\%$. The reason for the sharp decrease is that once the stability functions are active, they sanction unstable sequences by reduced quality values. They thereby exclude these sequences from S_Q and reduce the average set size nAv of the quality-optimal set S_Q. On the other hand, as the stability ratio is increased, more and more games allow no sequence changes at all and become completely stable. If a game becomes completely stable, all outcomes are identical and have the same quality regardless of the agents' strategy profiles. Thus in the case of complete stability S_Q covers the full set of Z.
- *Effects across stability functions:* The differences between SF_2 and SF_3 are comparatively small. For function SF_1, a significantly different behavior can be observed. Set sizes of S_Q do not reach 100% of Z at the upper end of the sr range. In fact, due to the characteristics of this stability function, complete stabilization of some games is not reached regardless of the value of the sr even for infinite sr.

7.2.5. Summary

The aim of section 7.2 has been to evaluate the average set sizes of the primary system-optimality sets S_H,S_P,S_T, and S_Q. These set sizes are important for the correct interpretation of the following results in subsection 7.3 and 7.4 which will be produced by intersecting the primary system-optimality sets with other system-optimality- and agent-rationality sets. Obviously, the size of any potential intersection depends on (and is upper-bounded by) the size of its primary sets.

Summarizing section 7.2, the following conclusions are drawn:

- The size of the Hicks-optimal set S_H is large (i.e. greater than 80% of the potential size) when the early-submission reward is inactive ($esr = 0$) and significantly reduced when it is active ($esr = 1$).
- The size of the Pareto-optimal set S_P is 100% of the potential size with the early-submission reward inactive ($esr = 0$) and small (below 3%) when its active ($esr = 1$).
- The time-optimal set S_T and its size are identical to Hicks-optimal set S_H computed with a setting of $esr = 0$. That is, it is also greater than 80% of the potential size.
- The quality-optimal set S_Q has a minimum size for the smallest non-zero stability ratio (here: $sr = 1\%$).
- The behaviors of (and results for) stability functions SF_2 and SF_3 are qualitatively similar, albeit with a slightly different sensitivity to parameter sr. The behavior of SF_1 deviates significantly, however. Most importantly, this is due to the fact that SF_1 does not establish a complete stabilization of the sequence regardless of the magnitude of the stability ratio sr.

7.3. Inner Consistency of Design Requirements for System-Optimal Sets

This subsection presents results regarding the inner consistency and compatiblity of the design requirements which were discussed in isolation in the previous section. The question to be answered is to which extent a mechanism variant M_i itself allows to satisfy all stated requirements at the same time. In order to answer that question, the non-emptiness of secondary sets DVI_1 and DVI_2 is analyzed. These sets represent design-view intersections of the primary system-optimal sets S_H, S_T, S_Q and S_E.

The inner consistency of design requirements as expressed by DVI_1 and DVI_2 is important, because before one demands that a predicted rational strategy A_{WD} coincides with the multiple system-optimal sets S_x, the system-optimal sets in themselves must not be contradictory in the first place. Where the system-optimal sets

are in fact contradictory, this is actually independent of the agents' decisions (i.e. it's also not the agents' fault). The effect on further incentive-compatibility measures would have to be considered.

7.3.1. Design-View Intersections

$\mathbf{DVI_1}$

The set DVI_1 is the intersection of time-optimal set S_T with earliest move S_E (equation 6.15).

The primary sets S_T and S_E are both independent of esr, thus DVI_1 is independent of esr. Both S_T and S_E are non-empty by definition. Thus, where DVI_1 is empty, both primary sets are non-empty but disjoint. The set S_E is always of size 1, thus the maximum size of DVI_1 is also one node. The type of analysis employed is therefore a non-emptiness proof $pph(DVI_1)$, evaluating the percentage of scenarios Sc in set SC where DVI_1 is non-empty.

$\mathbf{DVI_2}$

The set DVI_2, is the intersection of the time-optimal set S_T with the earliest move S_E, Hicks-optimal set S_H and quality-optimal set S_Q (equation 6.16).

The sets S_T, S_E and S_Q are independent of the setting of parameter esr. The set S_H however depends on the esr-setting, making DVI_2 a function of the esr-setting, too. All these node sets are non-empty by definition. The set sizes for S_H, S_T, and S_Q have been discussed above in section 7.2. The size of S_E is always just one node, the maximum size of DVI_2 is therefore also bounded by one. The type of analysis employed is therefore a non-emptiness proof $pph(DVI_2)$, evaluating the percentage of scenarios in SC where DVI_2 is non-empty.

Results

The results for $pph(DVI_1)$ and $pph(DVI_2)$ are shown in Figure 7.10. The results regarding DVI_1 are represented as solid lines. The results regarding DVI_2 are represented as dash-dotted lines.

The following results are observed:

- Results for $pph(DVI_2)$ are identical for $esr = 0$ and $esr = 1$. Thus, while a theoretical influence of esr on the results exist via the Hicks-optimal set S_H, this dependency does not have any measurable effect for the given scenario and mechanism design space. In the Figure, only one set of three dash-dotted lines is thus visible (one line for each stability function) which applies for $pph(DVI_2)$ with early-submission reward esr active or inactive.
- The results show that the non-emptiness proof $pph(DVI_1)$ holds for 87% to 100% (solid lines) and $pph(DVI_2)$ for 86% to 100% (dash dotted lines) of

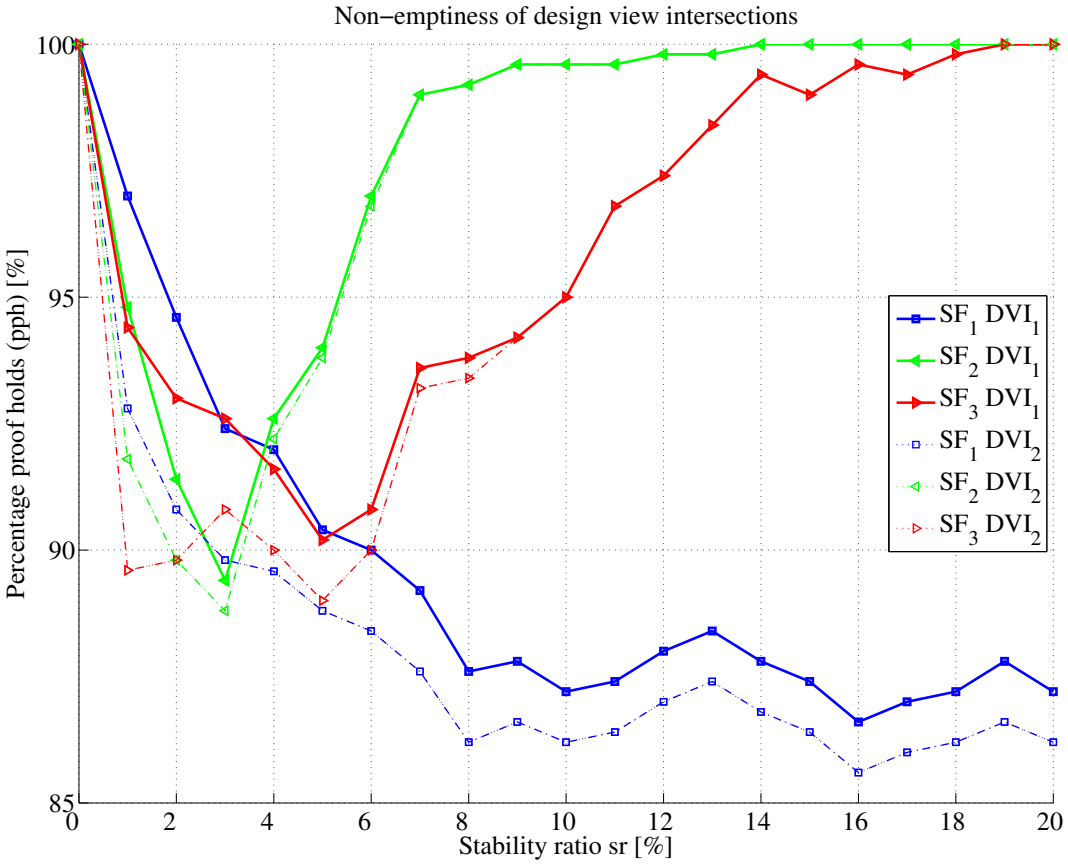

Figure 7.10.: Non-emptiness proof for design-view intersections, measured as percentage $pph(DVI_1)$ and $pph(DVI_2)$ - analyzing the percentage of scenarios where design requirements are compatible among themselves

all scenarios, depending on stability function SF and stability ratio sr. The characteristic curves are qualitatively similar to the sizes of time-optimal set S_T in Figure 7.8. For both design-view intersections, SF_2 and SF_3 reach the maximum possible values at both ends of the range for sr. However, stability function SF_1 shows a different characteristic. Here, values for $pph(DVI_1)$ and $pph(DVI_2)$ are the lowest at the upper end of the sr range.

- Results for $pph(DVI_1)$ are qualitatively similar to $pph(DVI_2)$ in the sense of some parallels between the curves for the respective stability function. Some differences exist, however. Depending on the stability function and stability ratio, DVI_2 can be up to 5 percentage points lower than DVI_1. This shows that a percentage of games exist in which a time-optimal solution is compatible with the earliest move (where all agents submit instantaneously), but incompatible with either the quality-optimal set S_Q or the Hicks-optimal set S_H.

State space representation of game with empty DVI_1

To illustrate a specific case of a game with an empty design-view intersection, Figure 7.11 shows a fraction of an example game where S_E is incompatible with S_T, thus DVI_1 is empty. The game is defined by the mechanism M_i with $sr = 3\%$, $esr = 1$, SF_1, and scenario Sc_{50}. The game has the earliest move $S_E = \{N1222\}$ and the time-optimal solutions $S_T = \{N1220, N1219, N1211, N1210\}$. The time-optimal nodes in S_T feature a process time of $PT = 168$ and a sequence outcome of ACDB while the earliest move S_E yields a process time of $PT = 178$ (thus not time-optimal) for the sequence outcome of ACBD.

Note that in outcomes of the time-optimal set, aircraft C delays the submission of the ETA-shift and achieves to influence the sequence. However, C itself does not profit but suffers from this strategy as no position gain is made but the early-submission reward is lost. Thus C's individual utility in S_T is lower than in S_E. However the joint utility J in S_T is higher than in S_E. Due to the fact that the delay is not individually profitable for aircraft C, (which is the actor in the position to choose), S_E turns out to be a Nash Equilibrium NE, while the outcomes represented in S_T are no NEs. Note that similar examples with empty DVIs can be found for all three stability functions.

7.3.2. Summary

In subsection 7.3 results have been presented regarding the inner consistency and compatibility of design requirements in order to see to a given mechanism variant M_i allows to satisfy all stated requirements at the same time.

In the majority of cases, the required inner consistency formulated by DVI_1 and DVI_2 is accomplished. However, assuming that a medium stability-ratio in the range $sr = [1\%, 10\%]$ is a likely choice for a final mechanism, inconsistencies and contradictions between the different design requirements of up to 15% of all scenarios exist. Thus, even when unrestricted cooperation of participating agents is assumed (i.e. willingness and ability to perform the behavior desired by the system designer), not all design requirements can continuously be fulfilled at the same time. The effect of this finding has to be taken into account in the interpretation of incentive-compatibility in the following (section 7.4). The effect is independent of - and cannot be reduced by- the activation of the early-submission reward esr.

A comparison of the results for DVI_1 and DVI_2 indicates that the majority of failed non-emptiness proofs for the $DVIs$ goes back to the demand to intersect time-optimal nodes S_T with the early move S_E (empty DVI_1). This means that S_E is occasionally incompatible with reaching a time-optimal solution S_T. The additional demand established by DVI_2 for the solution to be consistent with S_H and S_Q has a smaller effect on the results.

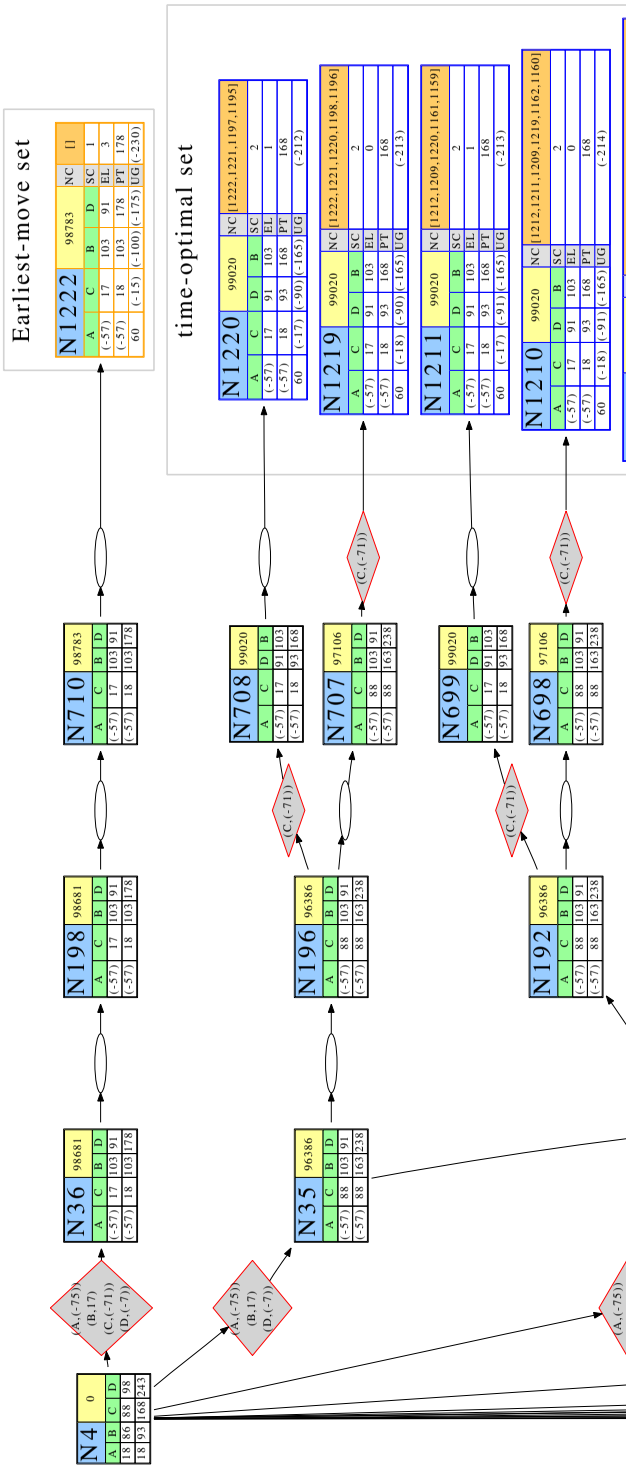

Figure 7.11.: Incompatibility of earliest-move S_E with time-optimal solution S_T (mechanism variant: SF_1, $sr = 3\%$, $esr = 1$, scenario: Sc_{50})

7.4. Compatibility of Design Requirements and Incentives for Agents

This section presents results on the core question of the analysis, that is the compatibility of the design requirements with incentives for agents and associated rational agent behavior. The results in this section are central to the prediction of overall system performance. They guide the developer's judgement whether the desired global system performance can be reached with agents striving to optimize their individual utilities by rational decisions.

Up to now, the two perspectives of agent-rationality and system-design view have been treated in isolation.

- Section 7.1 has presented results only on the agent-view.
- Sections 7.2 and 7.3 have discussed data only from the viewpoint of system-optimality.

In the following the confrontation of the two views is realized. The goal of that confrontation is to assess the likelihood that agents will act in line with the designer's expectations while they are trying to realize their individual goals.

Analysis Approach

The analysis is based on a number of formal proofs testing the non-emptiness of an intersection between the rational set A_R and different system optimal sets S_x (i.e. S_H, S_P, S_T, S_Q, S_E, DVI_1, DVI_2). The set intersections have been introduced in section 6.8.1. For the given scenario set SC, the percentage pph of scenarios Sc is considered for which the non-emptiness proof holds (compare equation 6.19). Figures 7.12 to 7.16 show the results for pph plotted against the stability ratio sr for the three considered stability functions SF_1, SF_2 and SF_3. Further, the results for the early-submission reward inactive ($esr = 0$) and active ($esr = 1$) are opposed (by showing solid and dash-dotted result curves respectively) wherever this reward has an effect.

In the following, each of the intersection-proofs is first regarded in an individual subsection. An overall summary and discussion of the significance of the results is given in subsection 7.4.7.

7.4.1. Intersection of Rational Set With Hicks-Optimal Set

This proof concerns the non-emptiness of an intersection of rational set A_{WD} with Hicks-optimal set S_H. Thus, the percentage of scenarios is calculated where the rational play of agents is compatible with the realization of a solution with maximum joint utility (equation 6.17).

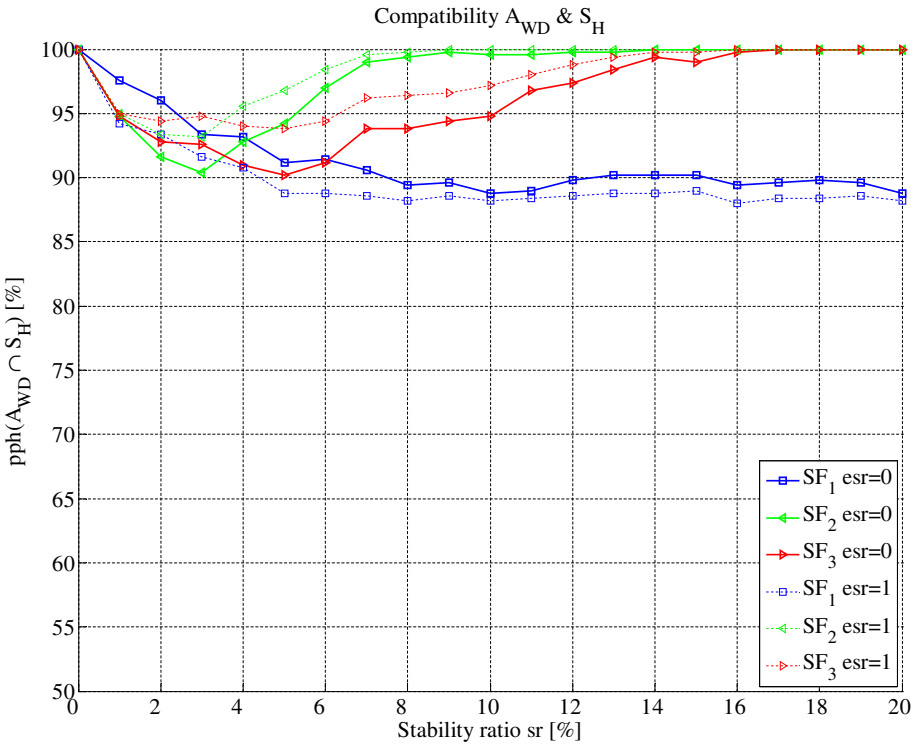

Figure 7.12.: Proof of non-emptiness for intersection of rational set A_{WD} and Hicks-optimal set S_H - analyzing percentage of scenarios where rational choices allow for outcome with maximum joint utility

In Figure 7.12 the results for $pph(A_{WD} \cap S_H)$ are illustrated. Results for the early-submission reward inactive ($esr = 0$) and active ($esr = 1$) are represented as solid and dash-dotted lines respectively.

The following effects are observed:

- *Compatibility level:* An non-empty intersection ($A_{WD} \cap S_H$) is found in 88-100% of the cases.
- *Effects across stability functions:* The characteristic curve corresponds to the pattern observed above. This implies that SF_2 and SF_3 reach intersections for 100% of the cases at both ends of the sr-range. Results for stability function SF_1 do not converge towards 100%.
- *Effects of early-submission reward:* Slight differences exist between the results for the cases of $esr = 0$ and $esr = 1$. Notably, the system performs better for $esr = 1$ for the stability function SF_2 and SF_3 (dash dotted line above solid counterpart). However, the setting of $esr = 1$ impairs the performance for SF_1 (dash-dotted line below solid counterpart) .

In summary, for more than 88% of cases the non-emptiness proof ($A_{WD} \cap S_H$) holds.

This means that to realize a Hicks-optimal outcome with maximum joint utility is compatible with rational agent behavior. For a significant percentage of up to 12% percent of all games, however, a rational agent play as suggested by A_{WD} is not compatible with a Hicks-optimal solution S_H of maximum joint utility. For SF_2 and SF_3, the desired compatibility can be improved by the activation of the early-submission reward ($esr = 1$).

7.4.2. Intersection of Rational Set With Pareto-Optimal Set

This proof concerns the existence of an intersection of rational set A_{WD} with Pareto-optimal set S_P. Thus the percentage of scenarios is tested in which rational agent play is compatible with a solution where no agent can gain without any other agent loosing something (equation 6.17).

The following results are found:

- *Compatibility level:* Agents' rational choices as predicted by A_{WD} are compatible with a Pareto-optimal solution in S_P for 100% of all scenarios, regardless of SF, sr and esr (Since all values are 100% no Figure is shown).
- *Effects of early-submission reward:* A value of $esr = 1$ reduces the average Pareto-set size $nAv(S_P)$ and the average rational-set size $nAv(A_{WD})$ significantly (see subsections 7.2.2 and 7.1.2). However, a Pareto-optimal solution in S_P is always contained in A_{WD}.

In summary, rational behavior in A_{WD} always allows for a Pareto-optimal solution where no agent can gain without another agent loosing something.

7.4.3. Intersection of Rational Set With Time-Optimal Set

This proof concerns the existence of an intersection of rational set A_{WD} with time-optimal set S_T. Thus the percentage of scenarios is tested in which rational agent play is compatible with an outcome that consumes the minimum process time pt and that would allow the maximum runway throughput (equation 6.17).

In Figure 7.13 the results for $pph(A_{WD} \cap S_T)$ are illustrated. Results for the early-submission reward inactive ($esr = 0$) and active ($esr = 1$) are represented as solid and dotted lines respectively.

The following facts are observed:

- *Compatibility level:* An non-empty intersection ($A_{WD} \cap S_T$) exists in 88-100% of the cases.
- *Effects across stability functions:* The result pattern is similar to the analysis of the intersection of A_{WD} and Hicks-optimal set S_H. The known curve characteristics for SF_1, SF_2, and SF_3 apply.

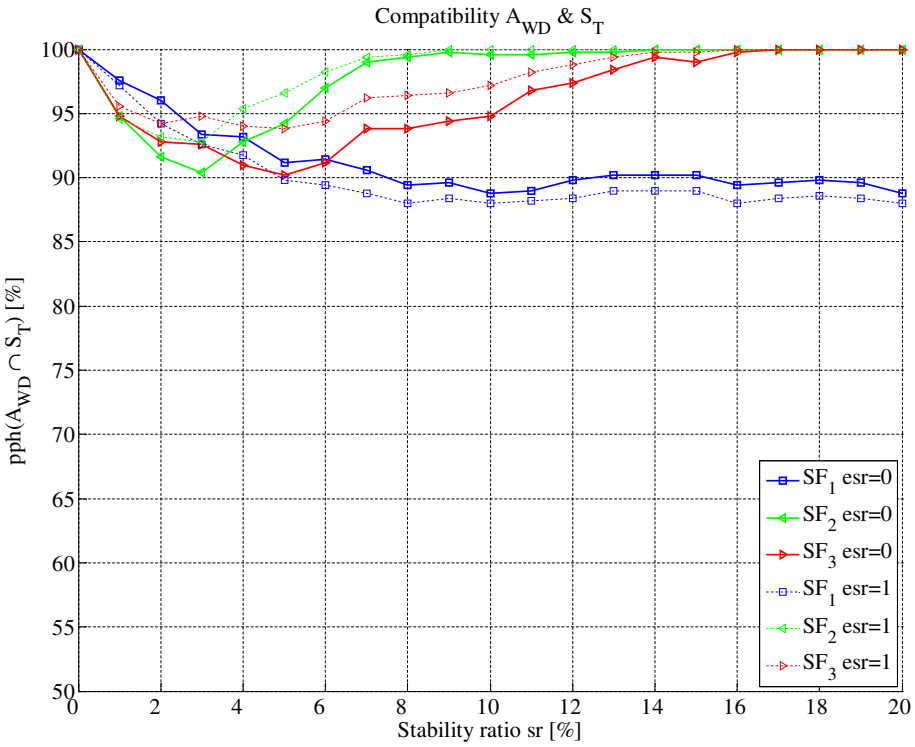

Figure 7.13.: Proof of non-emptiness for intersection of rational set A_{WD} and time-optimal set S_T - analyzing percentage of scenarios where rational choices allow for time-optimal outcome

- *Effects of early-submission reward:* Significant effects of the *esr*-setting are observed. For $esr = 1$ the system performance with stability functions SF_2 and SF_3 improves. Performance with stability function SF_1 deteriorates.
- *Relation to compatibility for Hicks-optimal set:* For the case of $esr = 0$, results for $pph(A_{WD} \cap S_T)$ and $pph(A_{WD} \cap S_H)$ are identical. Thus results represented here by solid lines are identical with their counterparts in Figure 7.12.

In summary, for more than 88% of cases the non-emptiness proof $pph(A_{WD} \cap S_T)$ holds. However, for a significant percentage of up to 12% of scenarios, a rational strategy profile as predicted by A_{WD} is incompatible with the realization of the time-optimal solution desired from the system-design view. This result is highly relevant from a global system performance point of view. It means that behavior which is rational from individual agents' point of view may lead to sub-optimal global system- performance in terms of runway throughput.

7.4.4. Intersection of Rational Set With Quality-Optimal Set

This proof concerns the non-emptiness of the intersection of rational set A_{WD} with quality-optimal set S_Q. Thus the percentage of scenarios is calculated in which rational agent play is compatible with a maximum-quality solution according the inner quality-criteria of the planning system. This solution is assumed to implement the will of the system designer for the optimal solution (equation 6.17).

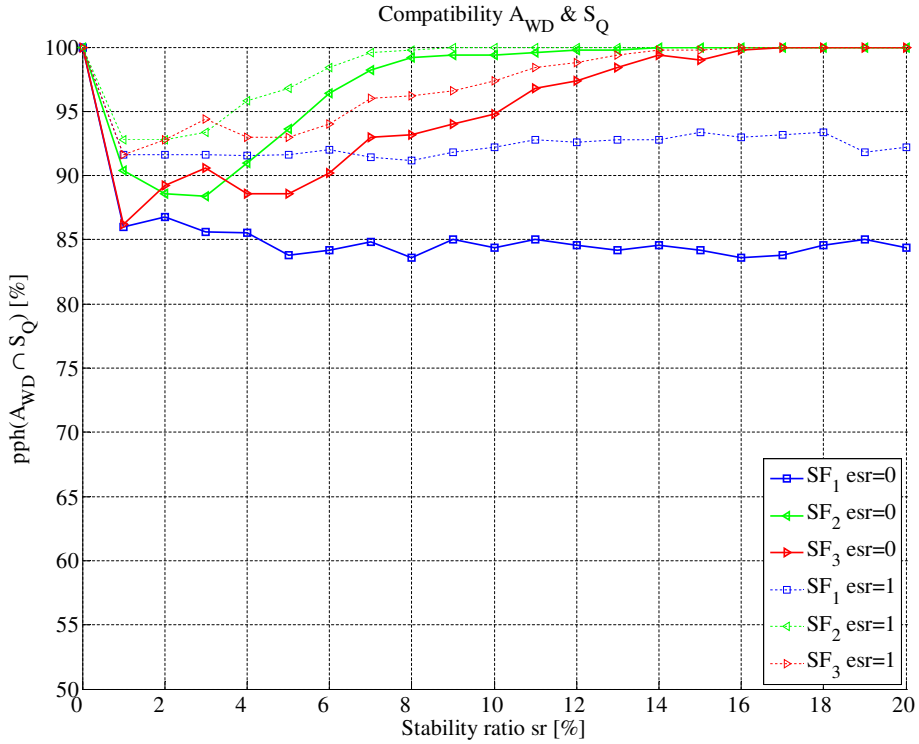

Figure 7.14.: Proof of non-emptiness for intersection of rational set A_{WD} and quality-optimal set S_Q - analyzing percentage of scenarios where rational choices allow for quality-optimal outcome

In Figure 7.14 the results for $pph(A_{WD} \cap S_Q)$ are illustrated. Results for the settings $esr = 0$ and $esr = 1$ are represented as solid and dotted lines respectively.

The following observations are made:

- *Effects across stability functions:* The patterns for $pph(A_{WD} \cap S_Q)$ of the three different SF repeat the characteristic pattern discussed above.
- *Effects of stability-ratio:* A sharper decrease is noted for lowest non-zero sr values (i.e. $sr = 1\%$ for the given discretization of the design space). This is explained by the size of the quality-optimal set S_Q (see Figure 7.9) which also has its minimum for this point of setting of the stability ratio sr.

- *Effects of early-submission reward:* A significant difference exists between the results for $esr = 0$ and $esr = 1$, with a notably larger quantitative impact than for the other dependent variables such as S_H and S_T. In contrast to the compatibility of rational set A_{WD} with Hicks- and time-optimal set, the value of $esr = 1$ here improves the performance for all three stability functions SF_1, SF_2, and SF_3 with the largest impact on SF_1.
- *Characteristics of* SF_1: Stability function SF_1 yields an almost constant level of compliance for all non-zero sr in the range $[1\%, 20\%]$.

It is concluded that for more than 85% of cases the non-emptiness proof $pph(A_{WD} \cap S_Q)$ holds (i.e. rational behavior does allow for a quality-optimal solution). However, for a significant percentage of up to 15% of scenarios, a rational strategy profile as predicted by A_{WD} is incompatible with the realization of a quality-optimal solution in S_Q, which represents the planning system's inner quality criteria and the assumed will of the system designer.

7.4.5. Intersection of Rational Set With Earliest-Move Set

This proof concerns the existence of an intersection of rational set A_{WD} with earliest-move S_E. Thus, the percentage of scenarios is tested in which it is rational for all agents to submit their modifications as early as possible during the first stage of the game, as desired from the global system's point-of-view (equation 6.17).

In Figure 7.15 the results for $pph(A_{WD} \cap S_E)$ are illustrated. Results for the settings $esr = 0$ and $esr = 1$ are represented as solid and dotted lines respectively.

The following observations are made:

- *Compatibility level:* For the case of early-submission reward inactive ($esr = 0$, solid lines) the pattern is similar to the intersection of $pph(A_{WD} \cap S_Q)$. However, here the absolute values go down to 81%, making this proof more selective.
- *Effects of early-submission reward:* For the case of $esr = 1$ (dotted lines) the situation is significantly improved. In more than 97% percent of all cases, the proof holds over the scenario set SC and the earliest-move constitutes a rational play according to the WD concept. The non-zero esr-value greatly increases the incentive to submit ETA-shifts early. This statement applies for all SF- variants and across the entire stability-range of sr.

In summary, for more than 80% of all cases the non-emptiness proof $pph(A_{WD} \cap S_E)$ holds. However, in up to 20% of the scenarios, it is against the interest of at least one agent to submit the ETA-shift during the first stage of the game. This is practically relevant because it means for certain scenarios that it is rational for an aircraft to hold back information in order to gain an advantage in the arrival sequence. Such behavior, if practiced systematically, would violate a fundamental behavioral requirement of the system design.

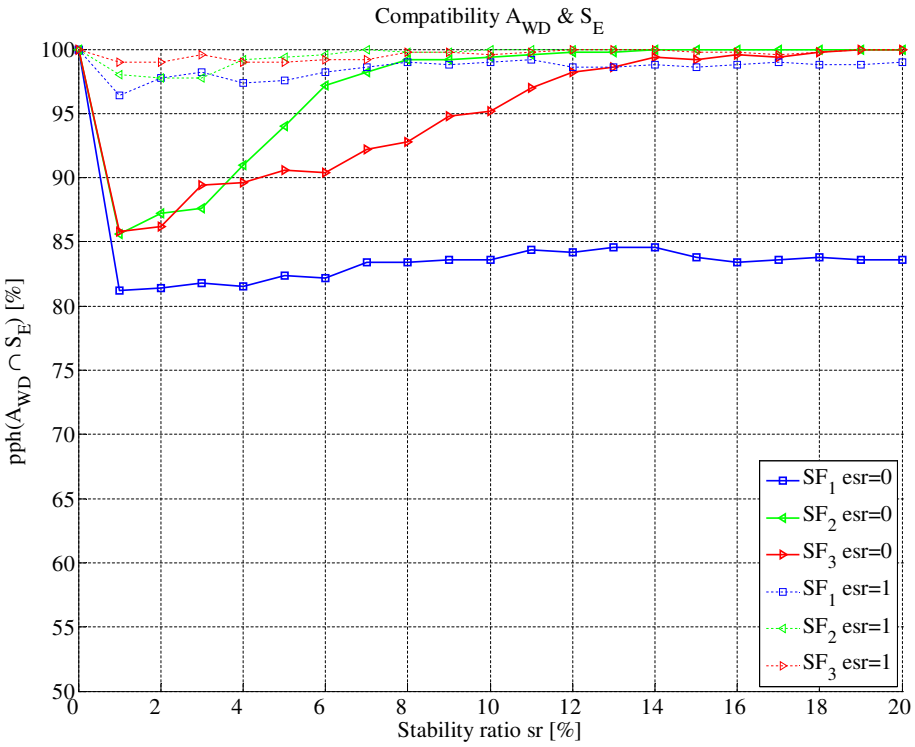

Figure 7.15.: Proof of non-emptiness for intersection of rational set A_{WD} and earliest-move S_E - analyzing percentage of scenarios where instantaneous submission of ETA-shifts is a rational choice

7.4.6. Intersection of Rational Set with Design-View Intersections

This proof concerns the existence of an intersection of rational set A_{WD} with design-view intersections DVI_1 and DVI_2. First, the percentage $pph(A_{WD} \cap DVI_1)$ is tested in which rational behavior is compatible with the earliest-move S_E and time-optimal behavior S_T at the same time (equation 6.18). Second, the percentage $pph(A_{WD} \cap DVI_2)$ of scenarios is tested in which rational behavior is compatible with time-optimal set S_T, earliest-move S_E, Hicks-optimal set S_H and quality-optimal set S_Q (equation 6.18).

In Figure 7.16 the results for $pph(A_{WD} \cap DVI_1)$ and $pph(A_{WD} \cap DVI_2)$ are illustrated. In the Figure, results concerning DVI_1 and DVI_2 are marked as solid and dotted lines respectively. As a second independent factor, results with early-submission reward inactive ($esr = 0$) and active ($esr = 1$) are marked separately in thick and thin linestyle.

The following results are observed:

- *Effects of stability functions:* For DVI_1 and DVI_2 the results show the same characteristic pattern with regard to all three function SF_1, SF_2, SF_3, as

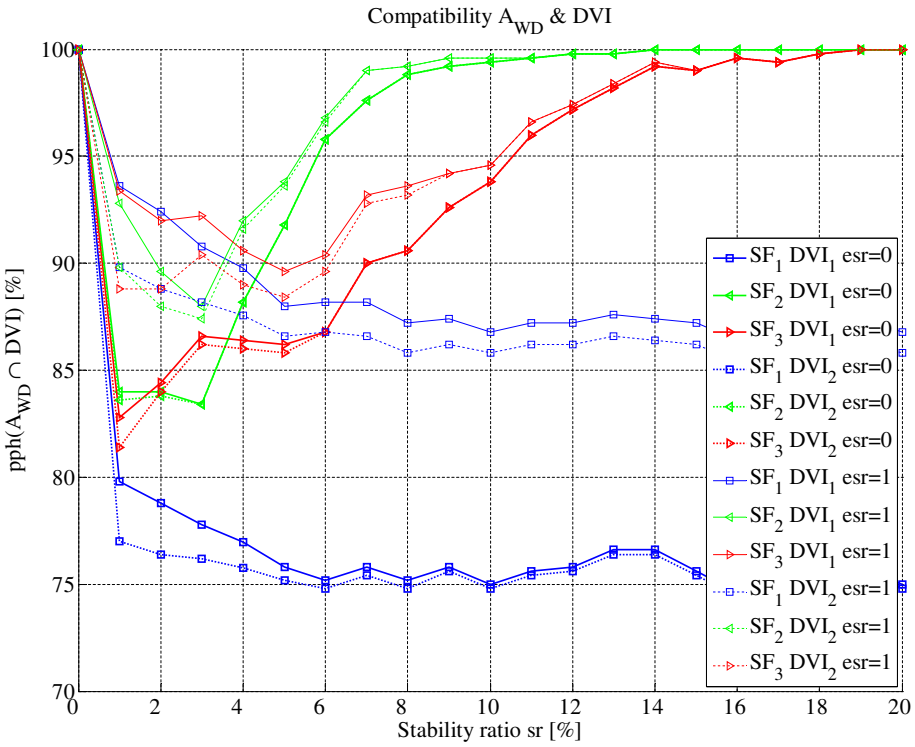

Figure 7.16.: Intersection proof of rational set A_{WD} and design-view intersections DVI_1 and DVI_2 - analyzing percentage of scenarios where rational choices allow for behavior which is compatible with design requirements

described above.

- *Effects of early-submission reward:* There is a significant positive influence of the setting of $esr = 1$ over $esr = 0$ for DVI_1 and DVI_2 and for all three stability functions SF_1, SF_2, and SF_3. This influence increases the percentage of games where A_{WD} is compatible with DVI_1 and DVI_2 (graphically, the thin line is above the thick line). For DVI_1, which is independent of the esr (see section 7.3.1), the effect has to be attributed entirely to the influence of esr on A_{WD}.

In summary, in between 75 and 100% of cases the non-emptiness proofs $pph(A_{WD} \cap DVI_1)$ and $pph(A_{WD} \cap DVI_2)$ hold (i.e. rational behavior is compatible with fulfilling the respective design requirements). But for up to 25% of cases the predicted agent-rational set A_{WD} is not compatible with DVI_1 or DVI_2, meaning that it would either be irrational for an agent to choose a behavior compatible with the DVIs or it would be impossible, because the DVIs themselves are empty for the specific game.

7.4.7. Summary

This section has presented exemplary results on the core question of the analysis:

How far do the mechanism variants reach alignment of incentives for agents (and resulting rational agent behavior) with the system requirements?

In general, none of the three stability functions SF_1, SF_2, or SF_3 is able to establish behavioral incentives which are 100% compatible with all formulated design requirements. The exceptions are the boundary cases of $sr = 0\%$ (where all stability functions are deactivated entirely) or sr-values beyond a point of complete stabilization (where sequence changes are totally suppressed regardless of agents' behavior). Both these extremes do not represent realistic options for a practical application of a sequence-planning mechanism.

In the relevant range of sr with an effective stability/performance trade-off between sr=0% and complete stability, some deficiencies regarding incentive-compatibility will have to be accepted for all considered mechanism variants in DS. Depending on the specific requirement and mechanism, up to 20% of the proofs have been shown to fail. The acceptance of such a level can be discussed in general. At the same time, the dependence of the results on sr and SF (i.e. the mechanism variant) points to a potential for optimizing the compatibility of incentives and design requirements by choosing a mechanism variant M_{opt} defined by an optimal combination of SF, sr and esr. The parameters SF, sr, and esr will thus be used for the optimization of the compatibility-results in conjunction with the quantitative performance metrics discussed in the following section.

7.5. Quantitative Analysis of Sequence Changes and Process Time

This section presents the results from the detailed quantitative evaluation of the inner set-characteristics of agent-rationality set and system-optimality sets.

Two different metrics are employed, which are important indicators and predictors of system performance and system workability. These are:

- The number of sequence changes $chAv$ (equation 6.21)
- The total process time $ptAv$ (equation 6.22)

Both metrics are applied to the set of agent-rational behavior A_{WD} and the system-optimal sets S_E and S_T, as they represent important behavioral requirements (S_E) and performance requirements (S_T) of the system design. The goal of the analysis is to illustrate and understand the consequences of the behavior described by these individual sets for the performance of the system as a whole.

Performance vs. stability trade-off

It is pointed out that the two metrics (sequence changes and process time) represent two inherently competing interests of the system design. In fact, the requirement of stability (operationalized as a low number of sequence changes) to a certain extent contradicts the requirement of minimizing process time. In the presence of dynamically changing situations (such as introduced by the shifting of ETAs), performance optimization is achieved by re-arranging aircraft in the sequence. Thus, an optimal trade-off between stability and performance has to be found, which weights both concerns according to their importance.

7.5.1. Sequence Changes

The graph in Figure 7.17 shows the results for the average number of sequence changes $chAv$ (equation 6.21) of outcomes in the considered agent rational set A_{WD} (solid lines), system optimal sets S_T (dash-dotted lines) and S_E (dotted lines). Due to the employed model and formalization of the arrival-management scenario as a four-stage game, the maximum theoretical number of sequence changes is four. This would mean the sequence changes for each stage of the game. The minimum number of sequence changes is zero, meaning that the initial sequence remains unchanged.

The following results are observed:

- *Range:* The average number of sequence changes lies in the range between 0 and approximately 1.42 changes per scenario. It depends on the set of outcomes (A_{WD}, S_T or S_E), the stability ratio sr and the stability function SF.
- *Effects across sets A_{WD},S_T and S_E:* When comparing the number of sequence changes across the sets, the order is generally $chAv(S_E) \leqslant chAv(A_{WD}(esr = 1)) \leqslant chAv(A_{WD}(esr = 0)) \leqslant chAv(S_T)$. Thus the earliest-move causes less sequence changes than the agent-rational moves, and these cause less than the time-optimal solution. This general order applies to all three stability functions SF.
  Between the agent-rational sets, the activation/deactivation of early-submission reward esr has a significant effect on the average number of sequence changes $chAv(A_{WD})$. For $esr = 1$ (thick solid line) the number of sequence changes is up to 50% lower than for $esr = 0$ (thine solid line) for any of the three stability functions SF. The system in general becomes much more stable for agent-rational behavior with the setting $esr = 1$.
  Generally values $chAv(A_{WD}(esr = 1)$ lie close to values for $chAv(S_E)$ while values for $chAv(A_{WD}(esr = 0))$ lie close to $chAv(S_T)$. This means that establishing the positive esr achieves roughly the same stability of the sequence as would be achieved by all agents playing their moves on the 1st stage of the game. On the other hand $esr = 0$ would result in approximately the number of sequence changes of the time-optimal solution (without necessarily realizing the benefits of the time-optimal solution).

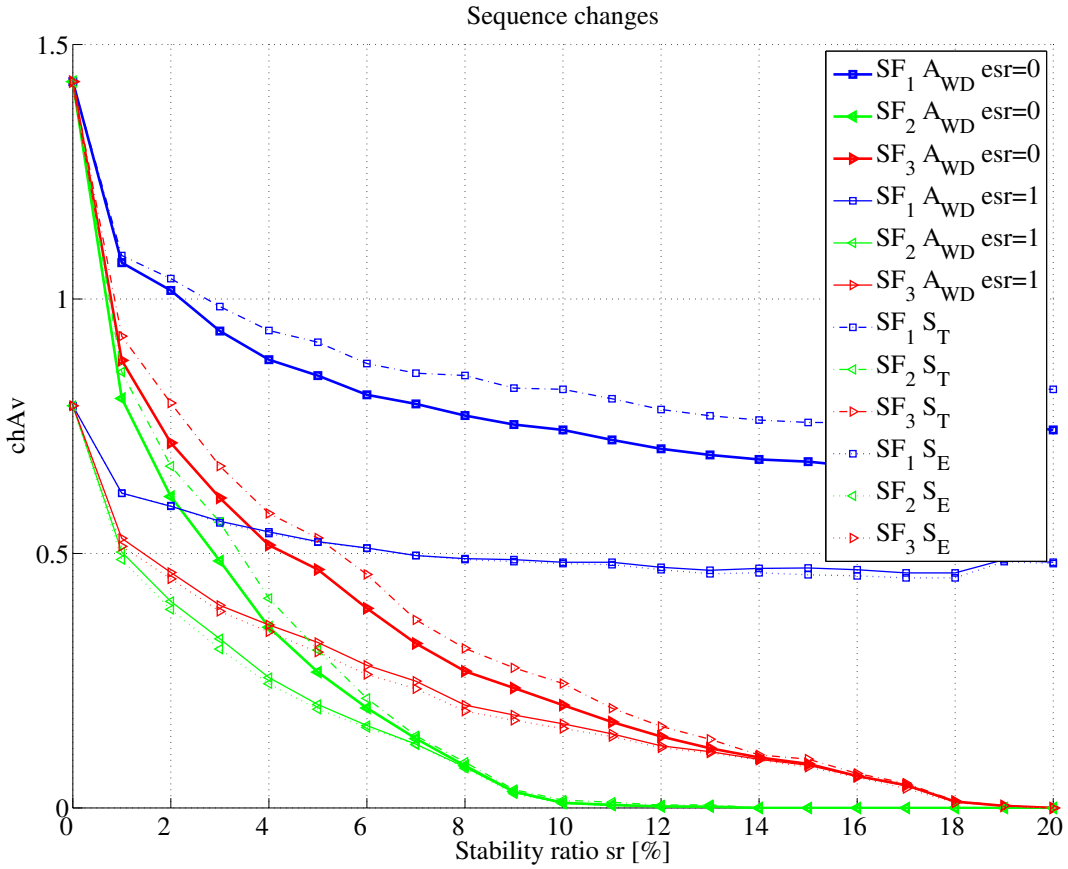

Figure 7.17.: Average number of sequence changes $chAv$ for rational set A_{WD}, time-optimal set S_T and earliest-move set S_E

- *Effects across stability functions:* When comparing SF_1, SF_2, and SF_3, the general order is $chAv(SF_2) \leqslant chAv(SF_3) \leqslant chAv(SF_1)$ for any given stability ratio sr. For SF_2 and SF_3 the system is completely stabilized at the upper end of the range of sr. This complete stability means that none of the scenarios in SC produces any sequence changes anymore, all outcomes in all games correspond to the respective initial sequence. For SF_1 complete stabilization is never reached, at least not in the considered range of sr.

7.5.2. Total Process Time

The graph in Figure 7.18 shows the results for the average process time (sum of all individual aircraft flight times) $ptAv$ (equation 6.22) of outcomes in the considered agent-rational set A_{WD} (solid lines),system-optimal sets S_T (dash-dotted lines), and S_E (dotted lines).

The following results are observed:

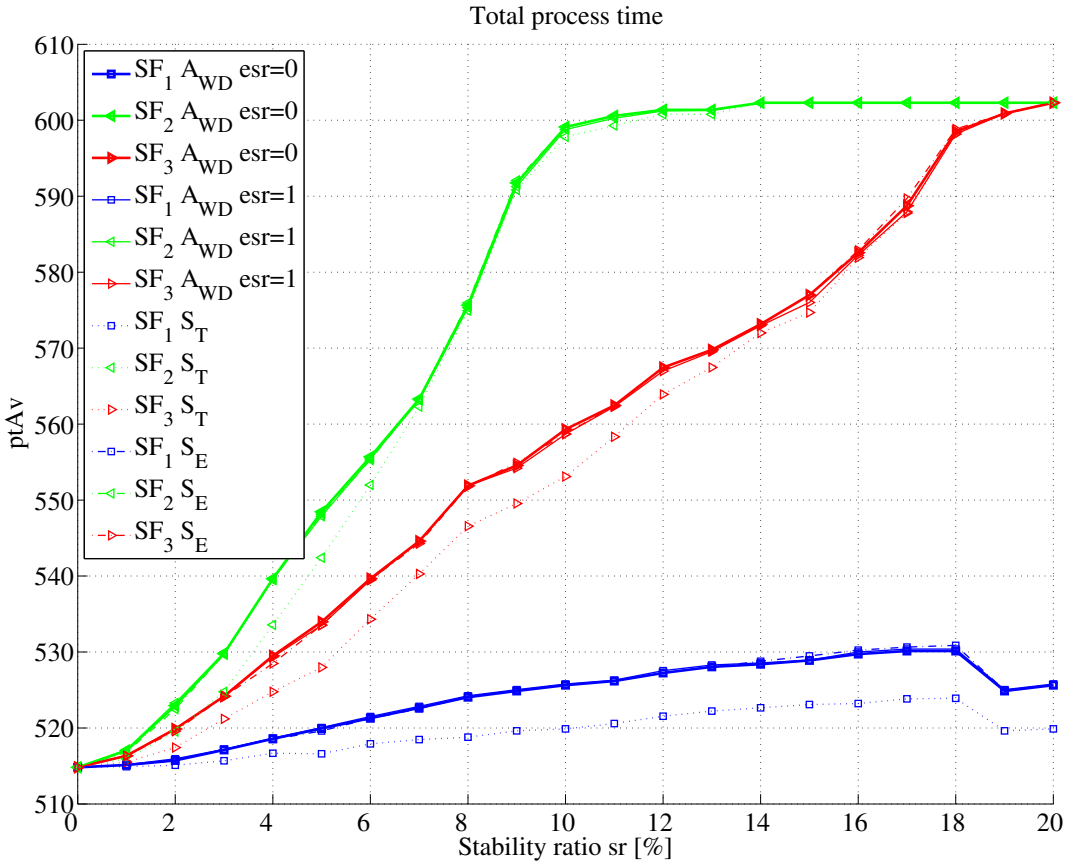

Figure 7.18.: Average process time $ptAv$ for rational set A_{WD}, time-optimal set S_T and earliest-move set S_E

- *Range:* The average process time lies in the range between approximately 515 and 605 seconds per scenario, depending on the set of outcomes (A_{WD}, S_T or S_E), stability ratio sr and stability function SF.
- *Effects across sets:* When comparing the average process times across the considered sets, the general order is $ptAv(S_T) \leqslant ptAv(A_{WD}(esr = 0)) \leqslant ptAv(A_{WD}(esr = 1)) \leqslant ptAv(S_E)$. While the time-optimal solution consumes less process time than the agent-rational moves, and these cause less than the earliest move, the differences between the last three sets are quantitatively small. That means that $A_{WD}(esr = 0)$, $A_{WD}(esr = 1)$, and $A_{WD}(S_E)$ have approximately the same cost in process time. Only the time-optimal solution stands out with a significant time gain.
- *Effects across stability functions:* When comparing SF_1, SF_2, and SF_3, the order is generally $chAv(SF_1) \leqslant chAv(SF_3) \leqslant chAv(SF_2)$. For SF_2 and SF_3 the system is completely stabilized (see section 7.5.1) and thus converges towards the process time needed when all sequences stay stable despite the submission

of the ETA-shifts. This process time is here approximately 605 seconds. For SF_1, complete stabilization is never reached, and sequence changes remain always possible. This has a positive effect on the process time needed.

7.5.3. Summary

Section 7.5 has presented results from the quantitative evaluations of inner set characteristics for the agent-rationality set and selected system-optimality sets. It has focused on the metrics of process time (i.e. sum of all individual flight times) and sequence stability (average number of sequence changes per scenario), which will be important to assess the performance of any potential mechanism variant. The following main effects haven been found, as could generally be expected:

- Sequence changes and stability ratio sr are negatively correlated.
- Process time and stability ratio sr are positively correlated.
- The better any mechanism $M = f(SF, sr, esr)$ performs in terms of process time, the higher the number of (generally undesirable) sequence changes tends to be.

While these main effects suggest qualitatively that there is an inherent conflict and trade-off in the requirements which cannot be extinguished entirely, the quantitative effects on the different performance metrics are of different strength. This fact generates a potential for optimization and the selection of an optimal mechanism M_{opt} for a given set of priorities, which will exploited below.

7.6. Selection of Optimal Mechanism Variant

In this section, the optimal mechanism variant $M_{opt} \in DS$ is selected based on the definition of design space DS and optimization criterion C_{ov}. The optimization assumes that agents play rational strategies as defined by the rationality set A_{WD}.

Optimization problem

Optimization criterion The optimization criterion C_{ov} combines the partial criteria incentive-compatibility, sequence changes, and process time. Adjustable weights reflect their relative priorities. The optimization criterion is defined as

$$C_{ov} = w_1 \cdot ic_{ov} + w_2 \cdot chAv + w_3 \cdot ptAv_{Dif} \; . \tag{7.1}$$

For the detailed definition of all factors see equation 6.28, section 6.9.

An optimal mechanism variant is selected from the design space DS by maximizing C_{ov} over the design space. For the analyis results presented below (Figure 7.19) the coefficient w_1 is set to a fixed value while w_2 and w_3 are varied so that

$$\begin{aligned} w_1 &= 5000 \ , \\ w_2 &= [0 ... -100] \ , \\ w_3 &= [0 ... -1] \ . \end{aligned}$$

This implies for the upper end of the ranges of w_2 and w_3:

- One second less process time (flight time summed up over all aircraft) is valued approximately equal to 0.01 sequence changes.
- One second less process time (flight time summed up over all aircraft) is valued approximately equal to 0.02 percent points gain in the incentive-compatibility metric.

By using the two degrees of freedom (DOF) of w_2 and w_3, their relative costs can be varied and adjusted with respect to value w_1 of overall incentive-compatibility.

Optimization points

For the actual selection of an optimal mechanism according to given cost function, three discrete optimization points are defined. These are characterized by the following combination of coefficients w_1,w_2 and w_3 with

$$\begin{aligned} P1 &= (w_1 = 5000,\, w_2 = -25,\, w_3 = -0.75) \ , \\ P2 &= (w_1 = 5000,\, w_2 = -50,\, w_3 = -0.50) \ , \\ P3 &= (w_1 = 5000,\, w_2 = -75,\, w_3 = -0.25) \ . \end{aligned}$$

From $P1$ over $P2$ to $P3$ the cost of sequence changes and instability in the sequence is valued increasingly higher in relation to the cost of additional process times.

Results

Graphical representation

In Figure 7.19 the result of the optimization are shown. The overall graph contains six subplots for the three stability functions SF_1, SF_2 and SF_3 (left to right) crossed with the two values for the reward for early submission $esr = 0$ (upper row) and $esr = 1$ (lower row). Within each subplot, the variation of the coefficients w_2 and w_3 are represented on the horizontal and vertical axis respectively.

The Figure content is interpreted as follows:

- The colored areas in each subplot mark combinations of coefficients w_1 and w_2 for which a mechanism with the respective stability function and esr-setting produces an optimal quality value C_{ov}.
- The color of the surface represents the corresponding stability ratio sr in the range of 1% to 20% (see colorbar in upper left subplot) which must be applied for this optimal mechanism variant.
- Where various stability ratios produce the same result (value of C_{ov}), the smallest optimal value of sr is given.
- In areas where no colored surface is presented (white background areas), the concerned mechanism-variant (combination of SF and esr) is strictly inferior to at least one other mechanism (i.e. not optimal). This combination should therefore not be selected.
- Where several subplots (combinations SF and esr) have colored surfaces for the same combination of w_2 and w_3, this means that all present optimal solutions with respect to C_{ov} with their respective stability ratios sr indicated by the surface color.

Qualitative results In summary, all subplots but the one for the combination $SFtype = SF_1$, $esr = 0$ (upper left subplot) contain optimal solutions for some combination of cost coefficients w_2 and w_3 together with an appropriate stability ratio sr.

Considering the influence of the esr-setting separately, the effect of setting $esr = 1$ is clearly positive. The mechanism variants which share this property (lower row) quantitatively cover wider ranges of coefficient combinations (greater versatility). In particular, they cover all the ranges where variants with $esr = 0$ can be optimal, too.

$\mathbf{M_{opt}}$ for optimization points $\mathbf{P_x}$ Optimization points $P1$, $P2$, and $P3$ are indicated by dashed lines in Figure 7.19. Where a subplot contains an optimal solution for this optimization point, this is marked by a label, indicating the point P_x and respective stability ratio sr. The following optimal mechanism solutions M_{opt} for the three points are found with

$$
\begin{aligned}
M_{opt}(P1) &= (SFtype = SF_3, esr = 1, sr = 1\%) \,, \\
M_{opt}(P2) &= (SFtype = SF_3, esr = 1, sr = 3\%) \,, \\
M_{opt}(P3) &= (SFtype = SF_2, esr = 1, sr = 8\%) \,.
\end{aligned}
$$

These are the mechanism variants which should be adopted by the designer for the respective optimization points.

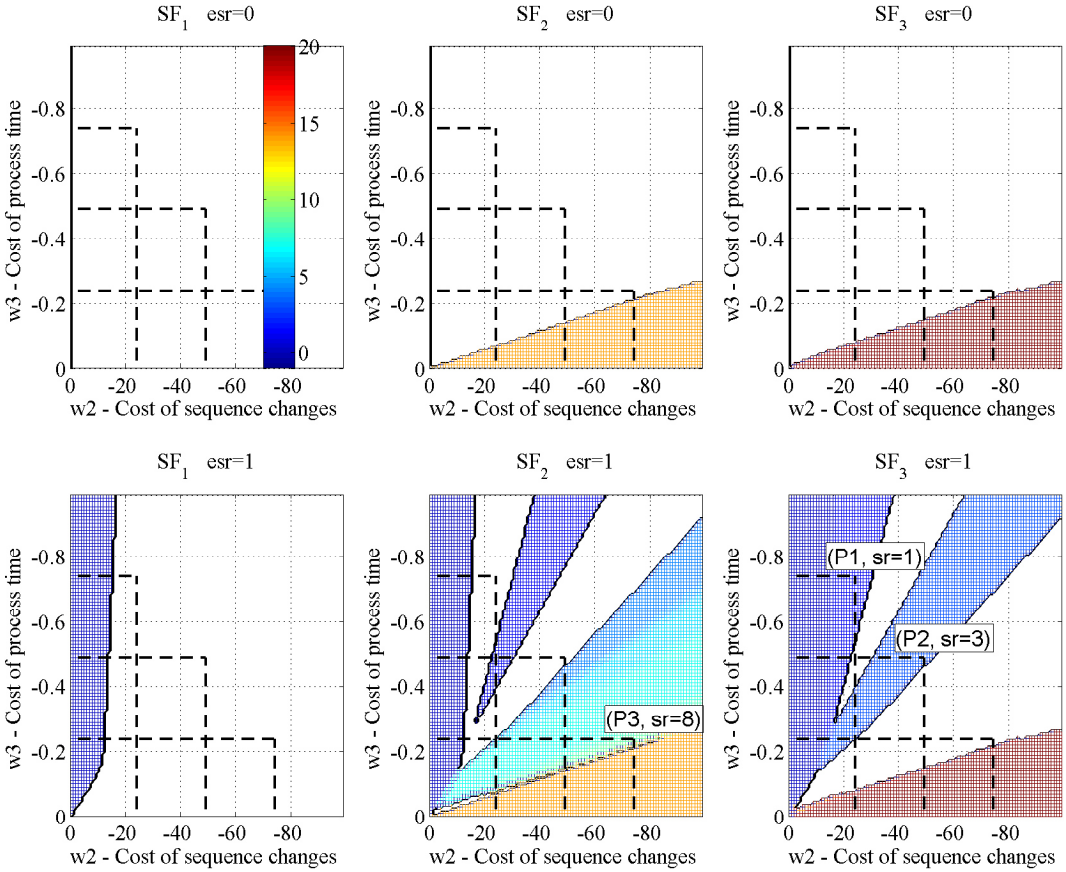

Figure 7.19.: Selection of optimal mechanism M_{opt}(with stability function type $SFtype$, stabilty ratio sr and early-submission reward esr) assuming rational agent behavior A_{WD} and varying cost coefficients w_2 (cost of sequence changes) and w_3(cost of time loss).

7.7. Effects on Incentive-Compatibility, Process Time, and Sequence Stability

This section considers the emergent system behavior which $M_{opt}(P1)$, $M_{opt}(P2)$, and $M_{opt}(P3)$ are expected to produce. Effects are shown in terms of

- resulting incentive-compatibility for the individual incentive-compatibility criteria (section 6.8.1, equations 6.17 and 6.18),
- resulting process times, and
- resulting sequence changes.

7.7.1. Effects on Incentive-Compatibility

In Figure 7.20 the differences regarding incentive-compatibility between the optimal mechanism variants are shown for $M_{opt}(P1)$, $M_{opt}(P2)$ and $M_{opt}(P3)$. In the grouped bar chart, each triple of bars represents one incentive-compatibility criterion discussed in subsections 7.4.1 to 7.4.6.

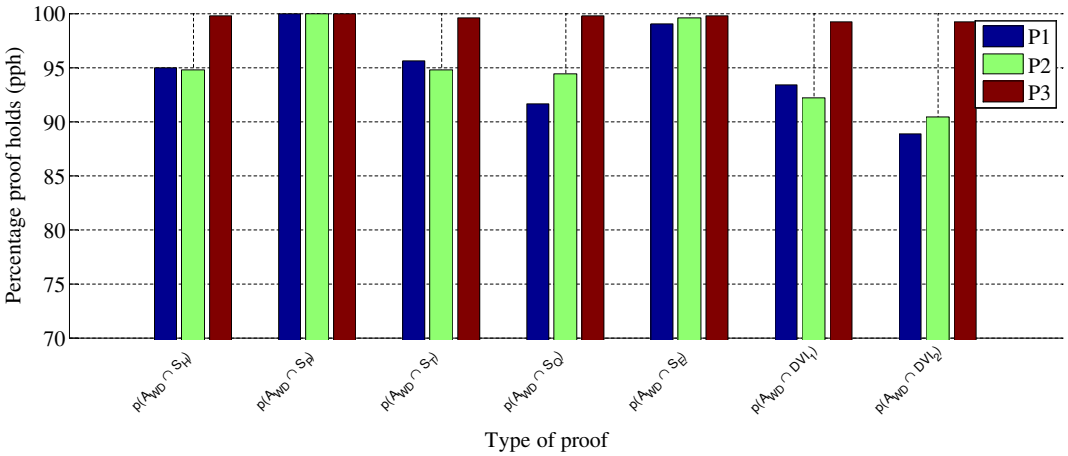

Figure 7.20.: Effects of different optimization-parameters on incentive-compatibility measures

The Figure shows that the best result in terms of incentive-compatibility is reached for mechanism variant $M_{opt}(P3) = \{SFtype = SF_2, esr = 1, sr = 8\%\}$. This mechanism variant performs best for six of seven criteria and as well as the others for the remaining one (compatibility with Pareto-optimal set S_P).

In a comparison of $M_{opt}(P1)$ and $M_{opt}(P2)$, the result is less obvious. Each variant fulfills three of the seven criteria better than the other and for one criterion the outcome is equal.

Overall, the percentages of cases for which the incentive-compatibility proof holds lie between 90% and 100%.

7.7.2. Effects on Sequence Changes and Process Time

Figure 7.21 shows the effects of the optimal mechanism variants $M_{opt}(P1)$, $M_{opt}(P2)$ and $M_{opt}(P3)$ on emergent system performance. In the left subplot, the three bars represent the average number of sequence changes for the optimal solutions at $P1$, $P2$ and $P3$. In the right subplot, the average process time $ptAv$ is indicated.

The results with regard to sequence changes are $chAv(M_{opt}(P1)) > chAv(M_{opt}(P2)) > chAv(M_{opt}(P3))$ and with view to process time $ptAv(M_{opt}(P1)) < ptAv(M_{opt}(P2)) < ptAv(M_{opt}(P3))$. Not surprisingly, this order mirrors the weighting of the respective

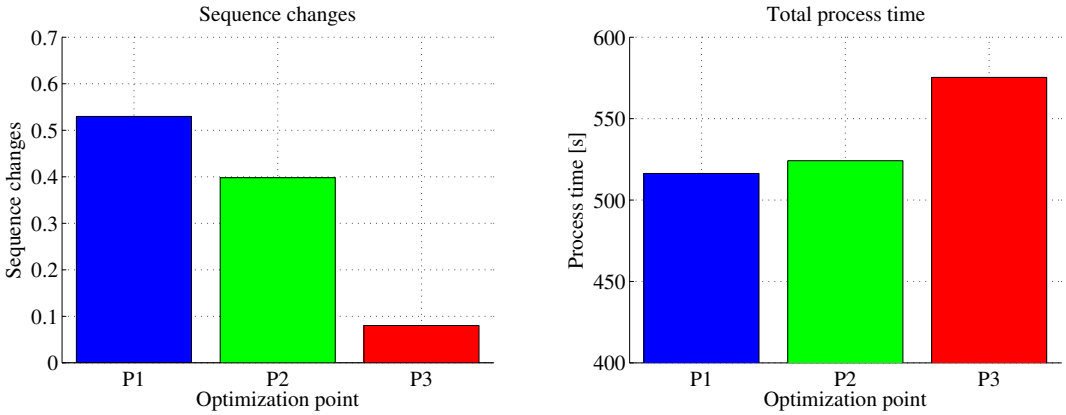

Figure 7.21.: Effects of M_{opt} at different optimization points $P1$,$P2$ and $P3$ in terms of sequence changes and process time

coefficients $w2$ and $w3$ for these solutions and reflects the inherent trade-off between stability and time-efficiency discussed above.

Quantitatively, the graph gives insights regarding the absolute number of sequence changes to be expected. These numbers range between 0.08 sequence changes for $P3$ and 0.52 sequence changes for $P1$ per scenario. It also shows the quantitative effect on process time in terms of the difference between the average times. Comparing $P1$ and $P3$, the difference is around 60 s, i.e. the difference is nearly as large as one landing slot.

7.8. Discussion

In chapter 7 the results of the application of the DAVIC framework to a class of mechanisms for cooperative arrival management have been presented. The main goal has been to select from design space DS of candidate sequence-planning mechanisms the mechanism-variant which combines high incentive-compability with high sequence-stability and low process time in an optimal way. The set-up of the analysis and optimization task were defined previously in chapter 6. In this chapter, the following main results have been found:

Prediction of rational choices: The predictive power of the four implemented agent-rationality solvers is compared in section 7.1. As a result, the most appropriate solver to be applied for the further analysis is selected. This is solver Sl_{WD} which implements the weak dominance concept (section 3.2.3). This choice is based on the credibility of the theoretical assumptions of this rationality-concept and on the ability of the solver Sl_{WD} to produce a solution for all scenarios and mechanism variants.

System-optimality sets: Subsequently the properties of the individual system-optimal sets have been analyzed (section 7.2) and the inner-consistency between the defined system-optimal sets have been studied by means of design-view intersections (section 7.3). One important finding of these analyzes is that inconsistencies and contradictions exist between some of the formulated design requirements. For example, the desired early submission of ETA-shifts is in some cases incompatible with a time-optimal planning solution. Overall, inconsistencies appear for up to 15% of the scenarios. These percentages depend on the mechanism variant M_i. Notably, this result implies that, regardless of how cooperative agents are, not all desirable properties can be fulfilled simultaneously for all scenarios.

Incentive-compatibility: Building on the individual results regarding agent-rationality sets (section 7.1) and system-optimality sets (section 7.2 and 7.3), it has been analyzed how far rational behavior is compatible with system-optimal behavior for mechanisms $M_i \in DS$ (section 7.4). This is a key analysis question within the DAVIC framework application, considering the overall goal of incentive-compatible design. The results reveal that rational choices are compatible with system-optimal behavior for the majority of scenarios (greater 75%) for all considered mechanism variants. Considerable differences exist between the mechanism variants, though. In general, stability functions SF_2 and SF_3 assure higher incentive-compatibility values than SF_1. Moreover, the activation of a direct reward for early submission proves clearly beneficial in terms of incentive-compatibility. Nevertheless, the results show that 100% incentive-compatibility can only be achieved for 'extreme' mechanism variants which are not credible for practical reasons.

Performance related to agent-rational and system-optimal behavior: The performance effects of agent-rational and system-optimal behavior have been studied in terms of process time and sequence stability (section 7.5). The results highlight the inevitable trade-off between these criteria and prove that improving the result for one criterion means impairing them for the other. However, the trade-off is not necessarily a zero-sum. There is potential for optimization with a given set of priorities.

Selection of optimal mechanism variant: The key task of the chapter has been to select a mechanism $M_i \in DS$ which combines incentive-compatibility, sequence stability, and process time in an optimal way (section 7.6). Optimal mechanism variants M_{opt} in terms of criterion C_{ov} have been chosen for three exemplary optimization points. Each optimization point represents a specific weighting of the partial criteria in C_{ov} and leads to a different optimal mechanism variant M_{opt} defined by its parameters stability function, stability ratio and early-submission reward. Comparing mechanism performance across the range of possible weights, it is constituted that mechanism variants SF_2 and SF_3, which rate sequence stability on the basis of position changes, perform better than SF_1, which rates stability on the basis of time changes. Also, the dedicated reward for early submission of ETA (with setting $esr = 1$) proves beneficial.

Performance effect per mechanism variant: The chapter closes with predictions (sec-

tion 7.7) of emergent system performance for mechanism variants M_{opt} at the three optimization points in terms of expected number of sequence changes and process time, assuming that agents make rational choices.

8. Summary and Outlook

In distributed systems and networks, such as the Air Traffic Management (ATM) system, peformance goals can only be reached if all actors cooperate in a coordinated way. In Air Traffic Management (ATM), in order to satisfy the future performance- and capacity-demands, increasingly sophisticated coordination procedures and resource allocation mechanisms are currently under development. These new mechanisms rely on extended data exchange as well as communication of user-preferences and -constraints between actors with entirely new negotiation protocols.

If the new mechanisms are designed properly, they ensure that Collaborative Decision Making (CDM) benefits can be realized and emergent system performance is indeed increased.

There is a risk and negative side-effect, however, that the new mechanisms may be more vulnerable to selfish, uncooperative behavior and manipulation than was previously the case. That kind of behavior can impair system performance.

To make sure that CDM benefits can in fact be realized, system design has to take care that incentives established by the system are correctly aligned (i.e. compatible) with the design goals. The assumption of cooperative behavior becomes much more credible, if it can be proven that cooperative interaction with a given mechanism is actually in the best interest of all participating agents.

Regarding the importance of incentives in the design of distributed systems, the introductory section of this thesis framed the following research questions:

- How can the importance of incentives be reflected in the design of cooperative multi-agent mechanisms?
- What kind of tools are needed to design mechanisms which fulfill the criteria of incentive-compatibility?
- How can the alignment of individual goals and global system goals be achieved through systematic engineering of suitable mechanisms?

Contributions to these questions have been developed and presented as part of this thesis on the levels of methods, tools and application.

8.1. Summary

The main contributions of this thesis are as follows:

- *Incentive-compatibility perspective:* Based on an analysis of the system engineering process practiced in the Air Traffic Management domain, the thesis demonstrates a mismatch between dominant ATM system characteristics, applied development paradigms, and prevalent engineering approaches today. It is shown that today's design practice assumes a cooperative attitude of agents as given. The design process regularly fails to prove that the system designer's notion of what actors *should* do, is also *rational* to do from the perspective of a self-interested agent. In view of these shortcomings, this thesis proposes to add the analysis of incentives (and incentive- compatibility) as a new perspective to system design and validation. The new perspective needs to be supported by appropriate methods & tools.
- *DAVIC framework development:* Within this thesis the novel framework DAVIC for the Design And Verification of Incentive-Compatible ATM systems is developed. This framework is based on decision-theoretic and system-theoretic analysis criteria as well as net-based modeling of distributed, concurrent ATM-systems. The core of the framework is a modeling approach based on Coloured Petri Nets (CPNs). By means of state space generation from CPNs with a method specifically developed, extensive decision spaces describing agent- and system-reactions can be explored. By means of formal analysis of the state spaces, the explicit confrontation of system- and agent view is realized to contrast what is *desirable* with what is *rational*. Based on the analysis capabilities, the framework supports a systematic optimization procedure to select the optimal mechanism variant in terms of incentive-compatibility and system-performance from a set of candidate mechanisms. As part of this thesis, the eight generic components of the DAVIC framework are comprehensively described, which help to structure the analysis- and design-process. The DAVIC framework is able to support incentive-compatible design in different application domains.
- *Framework application & implementation for arrival management:* The practical use of the DAVIC framework is demonstrated on a design problem from Air Traffic Management (ATM). The example case studies a class of potential future mechanisms for sequence planning in arrival management. These future mechanisms aim at building optimized sequences by processing aircraft inputs which are not considered today. As a negative side-effect though, these mechanisms might encourage uncooperative and selfish behavior in certain traffic situations. For example, aircraft may profit from holding back updated time estimates from the central planning system. The presented analysis set-up therefore seeks to identify the mechanism candidate which minimizes such uncooperative behavior and optimizes sequence stability and process time. One by one, the meaning of each framework component in the context of the arrival management application is explained. The implementation of the system-model and agent-model for the application is practically demonstrated. Also, the formulation of system-optimality criteria and agent-rationality concepts

for the specific application example is presented. Although the model makes some simplifications, the arrival management application has high industrial relevance and indicates how the proposed DAVIC framework can be applied in practice.

- *Application results:* The results from the DAVIC framework application to the arrival management problem are presented in detail. Results include quantitative evaluations across the set of 500 test scenarios as well as formal verification results and in-depth studies of individual scenarios on tree-based representations of the decision situations for participating agents. Results are discussed on the one hand in relation to the theoretical expectations (from graph theory and decision theory). On the other hand, they are interpreted from point of view of the application. The most important result is the selection of mechanisms from the design space which are optimal in terms of incentive-compatibility, process time and sequence stability. For example, it is found that the considered stability mechanisms on the basis of aircraft positions are more advantageous than the time-based stability function considered. It is further shown by means of formal verification that the formulated desired behavior from the system-design point of view contains some contradictions. Not all desirable properties can be fulfilled simultaneously for all scenarios, regardless of the choices that actors make. Finally, the expected performance for selected optimal mechanism variants is analyzed.

The main novel contribution of this work is the developement of a framework for the design of incentive-compatible systems. This framework achieves a seamless integration of modeling support with decision-theoretic analysis tools and system-optimality criteria. The framework thus allows a model-based confrontation of desirable agent behavior with rational agent behavior in order to design mechanisms which align both views. The characteristics and the structure of the DAVIC framework enable a tight coupling of incentive-compatible design with state-of-the-art system engineering and requirement engineering methods, which to the author's knowledge is not supported by existing isolated approaches so far.

8.2. Outlook

The thesis indicates several opportunities for extensions to the framework and for further research work on the development of incentive-compatible systems, particularly in the following areas:

- *Agent-rationality concepts and system-optimality concepts:* As part of the DAVIC framework, this thesis has discussed and implemented selected formalizations of games and decision situations together with their related solution concepts and equilibrium concepts. However, the toolbox of game theory and decision theory contains several alternative and refined formalizations which can be realized (e.g. Bayesian games with incomplete information). These can bring

additional insights, depending on the requirements of specific application examples. The same is true for the side of system-optimality concepts, where the DAVIC framework implementation can easily be enriched with additional criteria (e.g. Kaldor-Hicks efficiency) without any structural changes.

- *Modeling approach:* Coloured Petri Nets have been successfully employed as a modeling tool in this work and have shown high expressive power combined with capability for formal state space analysis. Although computational performance has not been the central focus of this study, any performance improvements which could in the future be realized (with CPN Tools or other state space tools) will allow the processing of more complex models with even richer scenario sets. Apart from performance aspects, it has been mentioned that the Petri net family offers a wide range of specialized modeling formalisms, which may be considered depending on the application. A major project for the future would be to investigate the use of hybrid (discrete-continuous) nets to power the framework. This would generate many new options in terms of modeling mechanisms, but also plenty of open questions regarding the computation of action-spaces and their decision-theoretic treatment.
- *Search procedure for optimal mechanism:* The identification of the optimal mechanism is currently realized in DAVIC as a one-pass selection from a predefined design space. This approach is likely to reach its limits when the theoretical design space of candidate mechanisms becomes very large. Then not all combinations of mechanism variants and scenarios can be exhaustively tested. Future work should consider the dynamic generation and refinement of new mechanisms at run-time. For example, genetic algorithms for the creation of mechanism candidates may be integrated in the toolbox. This would take the DAVIC framework a further step towards the automatic design of incentive-compatible mechanisms.
- *Application examples:* The framework should be applied to new applications, from within the domain of ATM-systems as well as from other multi-agent domains where incentives for agents' cooperation play an important role. For the ATM domain, some of the Collaborative Decision Making (CDM) applications mentioned in chapter 2 are potential candidates, such as trajectory negotiations, slot allocations, flight cancellations, airport operations center (APOC) or other Airport CDM measures which are currently under development. These new applications will put the generic approach and the re-usability of the framework to a test and will point out possible additional requirements.
- *Empirical validation:* The DAVIC framework relies on model-based predictions of agents' actions based on formal principles of rationality. For any model-based approach, the confrontation of its predictions with empirical data is a major test of its validity. In chapter 3 some insights from literature regarding the confrontation of behavioral data with normative theory have already been given. For the further development of the work of this thesis, in a first step,

empirical data on agents' behavior may be gathered in controlled laboratory environments with human operators in order to be compared with the model predictions. In a second step, field data from real world systems (e.g. CDM applications in ATM) should be analyzed. The results of research along these lines will be indicators of the practical value and the maturity of the DAVIC framework for industrial applications.

Abbreviations

AM	Agent Behavior Model
AMAN	Arrival Management System
ANSP	Air Navigation Service Provider
AR	Agent Rationality Concept
ARI	Agent Rationality Interface
ATC	Air Traffic Control
ATFM	Air Traffic Flow Management
ATM	Air Traffic Management
BDT	Behavioral Decision Theory
CDM	Collaborative Decsion Making
CDTI	Cockpit Display of Traffic Information
CFMU	Central Flow Management Unit
ClSimSSA	Closed Loop Simulation and State Space Analysis
CNS	Communication, Navigation and Surveillance
CPN	Coloured Petri Net
CS	Callsign
DA	Decision Analysis
DAVIC	Design and Verification of Incentive-Compatibility
DS	Design Space
DT	Decision Theory
DVI	Design View Intersection
E-OCVM	European Operational Concept Validation Methodology
ETA	Earliest Time of Arrival
FAA	Federal Aviation Administration
FMS	Flight Management System
GoC	Game of Chicken
GT	Game Theory
HPN	Higher Petri Net
IC	Incentive-Compatiblity
ICAO	International Civil Aviation Administration
IEDS	Iterative Elimination of Dominated Strategies
IP	Interaction Protocol
MD	Mechanism Design
MM	Mechanism Model
NE	Nash Equilibrium
NextGen	Next Generation Air Transportation System

OC	System-Optimality Concept
PN	Petri Net
PT	Place/Transition Net
SESAR	Single European Sky Air Traffic Management Research
SF	Stability Function
SM	System Model
SOI	System-Optimality Interface
SPE	Subgame-Perfect Equilibrium
SS	State Space
SWIM	System-Wide Information Management
TMA	Terminal Manoeuvring Area
TTA	Target Time of Arrival
TTO	Target Time Over
WD	Weak Dominance
WNE	Weak Nash Equilibrium

Bibliography

[Arro51] K. Arrow. *Social Choice and Individual Values.* Wiley, New York, 1951.

[ATT 12] AT&T Research. "GraphViz Manual". 2012. http://www.graphviz.org/. last checked:, 22.07.2012.

[Auma87] R. J. Aumann. "Game Theory". In: J. Eatwell, M. Migate, and P. Newman, Eds., *The New Palgrave: A Dictionary of Economics*, pp. 460–482, Macmillan, London, England, 1987.

[Ball00] M. O. Ball, R. Hoffman, C.-Y. Chen, and T. Vossen. "Collaborative Decision Making in Air Traffic Management: Current and Future Research Directions". Tech. Rep., NEXTOR - National Center of Excellence in Aviation Operations Research, 2000.

[Ball03] M. O. Ball, G. Donohue, and K. Hoffman. "Auctions for the Safe, Efficient and Equitable Allocation of Airspace Systems Resources". In: P. Crampton, Y. Shoham, and R. Steinberg, Eds., *Combinatorial Auctions*, pp. 507–538, MIT Press, Cambridge, 2003.

[Bill09] J. Billington and C. Yuan. "On Modelling and Analysing the Dynamic MANET ON-Deman (DYMO) Routing Protocol". *Transactions on Petri Nets and Other Models of Concurrency - Lecture Notes in Computer Science*, Vol. 5800, pp. 98–126, 2009.

[Bins07] J. H. van Binsbergen and L. M. Marx. "Exploring Relations Between Decision Analysis and Game Theory". *Decision Analysis*, Vol. 4, pp. 32–40, 2007.

[Born97] G. Bornstein, D. Budescu, and S. Zamir. "Cooperation in Intergroup, N-Person, and Two-Person Games of Chicken". *Journal of Conflict Resolution*, Vol. 41, No. 3, pp. 384–406, 1997.

[Brin11] C. Brinton, C. Provan, S. Lent, T. Prevost, and S. Passmore. "Collaborative Departure Queue Management". In: *Ninth USA/Europe Air Traffic Management Research and Development Seminar (ATM2011)*, FAA/Eurocontrol, Berlin, 2011.

[Broh99] A.-P. Bröhl and W. Dröschel. *Das V-Modell. Der Standard in der Softwareentwicklung mit Praxisleitfaden.* Oldenbourg, 1999.

[Bund10] Bundesstelle für Informationstechnik. "V-Modell XT Bund, Teil 1: Grundlagen des V-Modells". Tech. Rep., 2010.

[Came03] C. Camerer. *Behavioral Game Theory: Experiments in Strategic Interaction.* Princeton University Press, 2003.

[Cane09] R. Canetti and A. Rosen. "Cryptography and Game Theory - Lecture 9 - Games with Incomplete Information: Subgame Perfect Equilibria". Tech. Rep., Blavatnik School of Computer Science,Tel Aviv University,Israel, 2009.

[Caro07] T. E. Caroll and D. Grosu. "A Strategyproof Mechanism for Scheduling Devisable Loads in Linear Networks". In: *IEEE Parallel and Distributed Processing Symposium, IPDPS 2007*, pp. 1–9, Long Beach, CA, 2007.

[Clem06] J. Clempner. "Modeling Shortest Path Games With Petri Nets: A Lyapunov Based Theory". *Int. J. of Applied Math. Comp. Science*, Vol. 16, No. 3, pp. 387–397, 2006.

[CPN 12] CPN Tools. "CPN Tools - Computer Tools for Coloured Petri Nets". 2012. http://cpntools.org/. last checked:, 22.12.2012.

[Dash03] R. Dash, D. C. Parkes, and N. Jennings. "Computational Mechanism Design: A Call to Arms". *IEEE Intelligent Systems*, Vol. 18, No. 6, pp. 40–47, 2003.

[Davi05] R. David and H. Alla. *Discrete, continuous, and hybrid Petri nets.* Springer, Berlin, 2005.

[Dela02] O. Delain and J. Florent. "Collaborative Decision Making - Improving Airport Operations Through CDM". Tech. Rep., Eurocontrol Experimental Center, Bretigny, France, 2002.

[Dela03] O. Delain and A. Payan. "CDM Stockholm Arlanda WP1". Tech. Rep., Eurocontrol Experimental Center,Bretigny, France, 2003.

[Euro04] Eurocontrol. "Operational Concept Validation Methodology (OCVM)". Tech. Rep., 2004.

[Euro10] Eurocontrol. "Long-Term Forecast: Flight Movements 2010-2030". Tech. Rep., Eurocontrol, Brussels, 2010.

[Euro11a] Eurocontrol. "European Operational Concept Validation Methodology - Volume 1". Tech. Rep., Eurocontrol, Brussels, Belgium, 2011.

[Euro11b] Eurocontrol. "European Operational Concept Validation Methodology - Volume 2". Tech. Rep., Eurocontrol, Brussels, Belgium, 2011.

[Euro11c] Eurocontrol. "Performance Review Commission - Performance Review Report covering the calendar year 2010 (PRR2010)". Tech. Rep., Eurocontrol, Brussels, Belgium, 2011.

[FAAE08] FAA/Eurocontrol Cooperative R&D Action Plan. "Operational Concept Validation Strategy Document". Tech. Rep., 2008.

[Farl99] T. C. Farley, R. J. Hansman, M. R. Endsley, K. Amonlirdviman, and L. Vigeant-Langlois. "The Effects of Shared Information on Pilot-Controller Situation Awareness and Re-Route Negotiation". In: *10th International Symposium on Aviation Psychology*, Columbus, Ohio, USA, 1999.

[Fede09] Federal Aviation Administration (FAA). "FAA's NextGen Implementation Plan 2009". Tech. Rep., Federal Aviation Administration, 2009.

[Fede12] Federal Aviation Administration (FAA). "Next Generation Air Transportation System (NextGen)". 2012. http://www.faa.gov/nextgen/. last checked:, 22.07.12.

[Fram12] Framework. "Cambridge Dictionary Online". 2012. http://dictionary.cambridge.org/dictionary/british/framework. last checked:, 22.07.12.

[Gall02] G. Gallasch, L. M. Kristensen, and T. Mailund. "Sweep-Line State Space Exploration for Coloured Petri Nets". In: *4th Workshop and Tutorial on Practical Use of Coloured Petri Nets and CPN Tools*, Aarhus, Denmark, 2002.

[Gamr09a] D. Gamrad, H. Oberheid, and D. Söffker. "Automated detection of human errors based on multiple partial state spaces". In: *6th Vienna Conference on Mathematical Modeling ond Dynamical Systems MATHMOD*, pp. 651–659, Vienna, Austria, 2009.

[Gamr09b] D. Gamrad and D. Söffker. "Implementation of a Novel Approach for the Simulation of Cognition based on Situation-Operator Modeling". In: *SICE International Conference on Instrumentation, Control and Information Technology*, pp. 1404–1410, Fukuoka, Japan, 2009.

[Gamr11] D. Gamrad. *Modeling, Simulation and Realization of Cognitive Technical Systems*. PhD thesis, University of Duisburg-Essen, Chair of Dynamics and Control, 2011.

[Gint05] H. Gintis. "Behavioral Game Theory and Contemporary Economic Theory". *Analyse & Kritik*, Vol. 27, No. 1, pp. 6–47, 2005.

[Gree97] S. M. Green, T. Goka, and D. H. Williams. "Enabling User Preferences Through Data Exchange". In: *Proceedings of AIAA Guidance, Navigation and Control Conference*, New Orleans, LA, 1997.

[Haas10] P. J. Haas. *Stochastic Petri Nets: Modeling, Stability, Simulation*. Springer, Berlin, 2010.

[Hann08] R. Hann. "Zeitbasiertes Anflugmanagement mit 4D-CARMA zur Unterstützung von Dual Threshold Operations". In: *Deutscher Luft- und Raumfahrtkongress*, Darmstadt, Germany, 2008.

[Hans00] J. R. Hansman. "The Effect of Shared Information on Pilot/Controller and Controller/Controller Interactions". In: *USA/Europe ATM R&D Seminar*, Napoli, 2000.

[Hans05] S. O. Hansson. "Decision Theory - A Brief Introduction". Tech. Rep., Royal Institute of Technology (KTH), Sweden, Department of Philosophy and History of Technology, 2005.

[Harp05] R. Harper. "Programming in Standard ML". Tech. Rep., Carnegie Mellon University,Pittsburgh, Pennsylvania, 2005.

[Hars83] J. C. Harsanyi. "Bayesian Decision Theory, Subjective and Objective Probabilities, and Acceptance of Empirical Hypothesis". *Synthese*, Vol. 57, pp. 341–365, 1983.

[Hass09] A. Hasselberg, H. Oberheid, and D. Söffker. "State-Space-Based Analysis of Human Decision Making in Air Traffic Control". In: *7th Workshop on Advanced Control and Diagnosis*, Zielona Giora, Poland, 2009.

[Howa66] R. A. Howard. "Decision Analysis: Applied decision theory". In: *Fourth International Conference on Operational Research*, pp. 55–71, John Wiley, New York, 1966.

[Huan10] Y.-S. Huang and T.-H. Chung. "Modelling and analysis of air traffic control systems using hierarchical timed coloured Petri nets". *Transactions of the Institute of Measurement and Control*, Vol. 33, No. 1, pp. 30–49, 2010.

[Huck93] V. Huck. *Petrinetz-basierte Beobachtung von Systemzuständen.* Vol. 93-27 of *Forschungsberichtsreihe der Deutschen Forschungsanstalt für Luft- und Raumfahrt*, Wiss. Berichtswesen des DLR, Köln, Germany, 1993.

[Hurw72] L. Hurwicz. "On Informationally Decentralized Systems". In: C. McGuire and R. Radner, Eds., *Decision and Organization*, North Holland, Amsterdam, 1972.

[Info12] Informatik-Uni-Harburg. "Petri Nets Tools Database". 2012. http://www.informatik.uni-hamburg.de/TGI/PetriNets/tools/quick.html. last checked:, 22.07.12.

[Inte01] International Civil Aviation Organization (ICAO). "Procedures for Air Navigation Services - Rules of the Air and Air Traffic Services - ICAO DOC 4444 / ATM 501". 2001.

[Inte05] International Civil Aviation Organization (ICAO). "Global Air Traffic Management Operational Concept Document - DOC 9854 AN458". Tech. Rep., 2005.

[Inte11] International Civil Aviation Organization (ICAO). "ICAO Flight Plan Form". 2011.

[Jens06] K. Jensen, S. Christensen, and L. M. Kristensen. "CPN Tools - State Space Manual". Tech. Rep., University of Aarhus, Departement of Computer Science, 2006.

[Jens07] K. Jensen, L. M. Kristensen, and L. Wells. "Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems". *International Journal on Software Tools for Technology Transfer*, Vol. 9, pp. 213–254, 2007.

[Jens09] K. Jensen and L. M. Kristensen. *Coloured Petri Nets: Modeling and Validation of Concurrent Systems.* Springer, 2009.

[Jens97a] K. Jensen. *Coloured petri nets : Basic concepts, analysis methods and practical use.* Springer, Berlin, 1997. Vol. 1.

[Jens97b] K. Jensen. *Coloured petri nets : Basic concepts, analysis methods and practical use.* Springer, Berlin, 1997. Vol. 2.

[John88] R. E. Johnson and B. Foote. "Designing reusable classes". *Jounal of Object-Oriented Programming*, Vol. 1, pp. 22–35, 1988.

[Join07] Joint Planning and Development Office (JPDO). "Concept of Operations for the NextGen Air Transportation System". Tech. Rep., 2007.

[Jong05] H. de Jonge, E. Tuinstra, and R. Seljée. "Outbound Punctuality Sequencing by Collaborative Departure Planning". In: *6th USA/Europe ATM Seminar*, FAA/Eurocontrol, Baltimore, MD, 2005.

[Jonk04] G. Jonker, J.-J. Meyer, and F. Dignum. "A Market Mechanism for Airport Traffic Control". In: *EUMAS European Workshop on Multiagent Systems*, Barcelona, Spain, 2004.

[Jonk05a] G. Jonker, J.-J. Meyer, and F. Dignum. "Towards a Market Mechanism for Airport Traffic Control". In: C. Bento, A. Cardoso, and G. Dias, Eds., *Lecture Notes in Artificial Intelligence*, pp. 500–501, Heidelberg, 2005.

[Jonk05b] G. Jonker, J.-J. Meyer, and F. Dignum. "Efficiency and Fairness in Air Traffic Control". In: *Proceedings of BNAIC*, pp. 151–157, Brussels, Belgium, 2005.

[Jonk07a] G. Jonker, F. Dignum, and J.-J. Meyer. "Achieving Cooperation among Selfish Agents in the Air Traffic Management Domain using Signed Money". In: *6th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*, Honolulu, Hawai, 2007.

[Jonk07b] G. Jonker, H. Hesselink, F. Dignum, and J.-J. Meyer. "Preventing Selfish Behaviour in Distributed Tactical Airport Planning". In: *7th USA/Eurpoe ATM R&D Seminar*, FAA/Eurocontrol, Barcelona, Spain, 2007.

[Jonk08] G. Jonker. *Efficient and Equitable Exchange in Air Traffic Management Plan Repair Using Spender-Signed Currency.* PhD thesis, Universiteit Utrecht, The Netherlands, 2008.

[Keys07] D. J. Keys and B. Schwartz. "'Leaky' Rationality, How Research on Behavioural Decision Making Challenges Normative Standards of Rationality". *Perspectives on Psychological Science*, Vol. 2, No. 2, pp. 162–180, 2007.

[Kohl06] M. Köhler. "Formalising Multi-Agent Organisations". In: G. Lindemann and H. Schlinghoff, Eds., *Concurrency, Specification and Programming CS&P*, pp. 104–115, Humbold University Berlin, Berlin, 2006.

[Kohl07] M. Köhler, R. Langer, R. von Lüde, D. Moldt, H. Rölke, and R. Valk. "Socionic Multi-Agent Systems Based on Reflexive Petri Nets and Theories of Self-Organisation". *Journal of Artificial Societies and Social Simulation*, Vol. 10, No. 1, 2007.

[Korn06] B. Korn, H. Helmke, and A. Kuenz. "4D Trajectory Managment in the Extended TMA: Coupling AMAN and 4D FMS for Optimized Approach Trajectories". In: *ICAS 2006 - 25th International Congress of the Aeronautical Sciences*, Hamburg, 2006.

[Kova05] A. Kovacs, E. Nemeth, and K. Hangos. "Modeling and Optimization of Runway Traffic Flow Using Coloured Petri Nets". In: *International Conference on Control and Automation ICCA*, pp. 881–886, Budapest, Hungary, Budapest, Hungary, 2005.

[Kris00] L. M. Kristensen. *State Space Methods for Coloured Petri Nets.* PhD thesis, University of Aarhus, 2000.

[Le05] L. Le, G. Donohue, and C.-H. Chen. "Using Auction-Based Slot Allocation for Traffic Demand Management at Hartsfield Atlanta International Airport: A Case Study". *Journal of the Transportation Research Board*, No. 1888, pp. 50–58, 2005.

[Liu07] L. Liu and J. Billington. "Verification of the Capability Exchange Signalling Protocol". *International Journal on Software Tools for Technology Transfer (STTT)*, Vol. 9, No. 3-4, pp. 305–326, 2007.

[Luce89] R. D. Luce and H. Raiffa. *Games and decisions : Introduction and critical survey.* Dover Publications, New York, 1989.

[Mart99a] P. Martin. "Report of the Ad-Hoc Expert Group on CDM". Tech. Rep., Eurocontrol Experimental Centre, Bretigny, France, 1999.

[Mart99b] P. Martin, A. Hudgell, S. Vial, N. Bouge, N. Dubois, H. De Jonge, and O. Delain. "Potential Applications of Collaborative Decision Making". Tech. Rep., Eurocontrol Experimental Center, Bretigny, France, 1999.

[Mohl07] C. Möhlenbrink, H. Oberheid, and B. Werther. "Model based work process design in air traffic control". In: *8th ONERA-DLR Aerospace Symposium (ODAS)*, Göttingen, Germany, 2007.

[Mohl10] C. Möhlenbrink and E. Schnieder. "Fast and Frugal Heuristics Conquer the Airport". In: *AHFE International*, CRC Press / Taylor & Francis, Miami, USA, 2010.

[Mohl11] C. Möhlenbrink. *Modellierung und Analyse von menschlichen Entscheidungsheuristiken mit farbigen Petrinetzen.* PhD thesis, Technische Universität Braunschweig, Inst. f. Verkehrssicherheit u. Automatisierungstechnik, 2011.

[Mold01] D. Moldt and H. Rölke. "Verhaltensmodellierung von Petrinetz-Agenten". In: H. Giese and S. Philippi, Eds., *Visuelle Verhaltensmodellierung verteilter und nebenläufiger Software-Systeme*, pp. 92–97, Universität Münster, 2001.

[Mold97] D. Moldt and F. Wienberg. "Multi-agent systems based on coloured Petri nets". In: P. Azema and G. Balbo, Eds., *Lecture Notes in Computer Science: 18th International Conference on Application and Theory of Petri Nets*, pp. 82–101, Springer, Berlin, 1997.

[Myer04] R. B. Myerson. "Comments on Games with Incomplete Information Played by 'Bayesian' Players, I-III - Harsanyi's Games with Incomplete Information". *Management Science*, Vol. 50, No. 12, pp. 1818–1824, 2004.

[Myer99] R. B. Myerson. "Nash Equilibrium and the History of Economic Theory". *Journal of Economic Literature*, Vol. 37 (3), pp. 1067–1082, 1999.

[Nash50] J. F. Nash. *Non-Cooperative Games.* PhD thesis, Princeton University, 1950.

[Neum28] J. von Neumann. "Zur Theorie der Gesellschaftsspiele". *Mathematische Annalen*, Vol. 100, pp. 295–320, 1928.

[Neum37] J. von Neumann. "Über ein ökonomisches Gleichungssystem und eine Verallgemeinerung des Brouwerschen Fixpunktsatzes". *Ergebnisse eines Mathematik Kolloqiums*, Vol. 8, pp. 75–83, 1937.

[Neum44] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behaviour.* Princeton University Press, Princeton, New Jersey, 1944.

[Neum90] F. Neumann and H. Erzberger. "Analysis of Sequencing and Scheduling Methods for Arrival Management". Tech. Rep., NASA, Ames Research Center, 1990.

[Nils03] J.-E. Nilsson. "Marginal cost pricing of airport use: The case for using a market mechanism for slot pricing". Tech. Rep., Swedish National Road and Transportation Research Institute, 2003.

[Nola10] M. S. Nolan. *Fundamentals of Air Traffic Control.* Delmar Cengage Learning, New York, 2010.

[Nori07] A. Norin, T. Andersson, P. Värbrand, and D. Yuan. "A GRASP Heuristic for route planning of de-icing vehicles at Stockholm Arlanda Airport". In: *6th Eurocontrol INO Workshop*, Eurocontrol, Bretigny, France, 2007.

[Ober05] H. Oberheid. *Entwicklung eines Software-Assistenzsystems zur Unterstützung kollaborativer Entscheidungsprozesse in einem Flughafenmanagement-Leitstand.* Diploma thesis, University of Duisburg-Essen, Chair of Dynamics and Control, 2005.

[Ober06] H. Oberheid. "A Coloured Petri Net Model of Cooperative Arrival Planning in Air Traffic Control". In: K. Jensen, Ed., *Seventh Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools*, pp. 177–196, University of Aarhus, Denmark, Aarhus, Denmark, 2006.

[Ober08a] H. Oberheid, D. Gamrad, and D. Söffker. "Closed Loop State Space Analysis and Simulation for Cognitive Systems". In: *8th International Conference on Application of Concurrency to System Design (ACSD)*, pp. 39–44, Xi'An, China, 2008.

[Ober08b] H. Oberheid and D. Söffker. "Cooperative Arrival Management in Air Traffic Control - A Coloured Petri Net Model of Sequence Planning". In: K. M. van Hee and R. Valk, Eds., *International Conference on Application and Theory of Petri Nets and Other Models of Concurrency - ATPN*, pp. 348–367, Xi'An, China, 2008.

[Ober08c] H. Oberheid, M.-M. Temme, A. Kuenz, V. Mollwitz, and H. Helmke. "Fuel Efficient And Noise-Reduced Approach Procedures Using Late Merging of Arrival Routes". In: *German Aerospace Congress 2008*, Darmstadt, Germany, 2008.

[Ober09] H. Oberheid and D. Söffker. "Model-Based Analysis of Agents' Incentives in a Distributed Air Traffic Management Process". In: *6th Vienna Conference on Mathematical Modeling on Dynamical Systems,MATHMOD*, pp. 512–523, Vienna, Austria, 2009.

[Ober10] H. Oberheid, A. Hasselberg, and D. Söffker. "Know Your Options - Analyzing Human Decision Making in Dynamic Task Environments with State Space Methods". In: D. de Waard, N. Gerard, L. Onnasch, R. Wiczorek, and D. Manzey, Eds., *Human Centered Automation*, pp. 285–300, Shaker, Maastricht, The Netherlands, 2010.

[Offi07] Official Airline Guide, OAG. "Statistical Review 2007". Tech. Rep., Bedfordshire,UK, 2007.

[Park01] D. C. Parkes. *Iterative Combinatorial Auctions.* PhD thesis, Department of Computer and Information Science, University of Pennsylvania, 2001.

[Petr62] C. A. Petri. *Kommunikation mit Automaten.* PhD thesis, Rhein.-Westf. Institut für Instrumentelle Mathematik , Universität Bonn, Germany, 1962.

[Prev03] T. Prevot, P. Lee, T. Callantine, N. Smith, and E. Palmer. "Trajectory-Oriented Time-Based Arrival Operations: Results and Recommendations". In: *5th USA/Europe Air Traffic Management Research and Development Seminar*, Budapest, Hungary, 2003.

[Prin12] Principia Cybernetica Web. "Web Dictionary of Cybernetics and Systems". 2012. last checked:, 22.07.2012.

[Raif02] H. Raiffa, J. Richardson, and D. Metcalfe. *Negotiation analysis : the science and art of collaborative decision making*. Belknap Press of Harvard University Press, Cambridge, Mass., 2002.

[Rani09] A. Ranieri and L. Castelli. "A Market Mechanism to Assign Air Traffic Flow Management Slots". In: *Eighth USA/Europe Air Traffic Management Research and Development Seminar*, FAA/Eurocontrol, Napa Valley, California, 2009.

[Rapo00] A. Rapoport and R. Zwick. "Game theory". In: A. E. Kazdin, Ed., *Encyclopedia of Psychology*, Oxford University Press, New York, 2000.

[Roya07] Royal Swedish Academy of Sciences. "Mechanism Design Theory: Scientific Background on the Sveriges Riksbank Prize in Economic Sciences in Memory of Alfred Nobel 2007". Tech. Rep., 2007.

[Ruck97] W. Ruckdeschel. *Modellierung regelbasierten Pilotenverhaltens mit Petrinetzen*. VDI Verlag, Düsseldorf, 1997.

[Schn92] E. Schnieder and D. Abel. *Petrinetze in der Automatisierungstechnik*. Oldenbourg Verlag, München, 1992.

[SESA] SESAR Joint Undertaking (SJU). "SESAR FactSheet - System Wide Information Management (SWIM)". Tech. Rep., Brussels, Belgium.

[SESA06a] SESAR Consortium. "D1 - Air Transport Framework - The Current Situation". Tech. Rep., Brussels, Belgium, 2006.

[SESA06b] SESAR Consortium. "D3 - The ATM Target Concept - SESAR Definition Phase - Deliverable 3". Tech. Rep., Brussels, Belgium, 2006.

[SESA08] SESAR Consortium. "Investigate Needs for New Appropriate Modelling and Validation Tools and Methodologies - SESAR Definition Phase - Task 2.3.2". Tech. Rep., Brussels, Belgium, 2008.

[SESA09] SESAR Consortium. "European Air Traffic Management Master Plan". Tech. Rep., Brussels, Belgium, 2009.

[SESA12] SESAR Joint Undertaking (SJU). "SESAR Joint Undertaking". 2012. http://www.sesarju.eu/. last checked:, 22.07.12.

[Shan98] A. Shantz. "Weather Avoidance Planning and Collaborative Routing Decisions". In: *2nd USA/Europe Air Traffic Management R&D Seminar*, FAA/Eurocontrol, Orlando, Florida, 1998.

[Shoh08] Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, UK, 2008.

[Shor12] M. Shor. "Subgame Perfect Nash Equilibrium". 2012. http://www.gametheory.net/Dictionary/SubgamePerfect.html. last checked:, 22.07.2012.

[Smit88] J. Smith. *Decision Analysis - A Bayesian Approach*. Chapman and Hall Ltd., London, 1988.

[Soff03] D. Söffker. *Systemtheoretische Modellbildung der wissensgeleiteten Mensch-Maschine-Interaktion*. Logos Verlag, Berlin, 2003.

[Sull03] A. Sullivan and S. M. Sheffrin. *Economics: Principles in action*. Pearson Prentice Hall, Upper Saddle River, New Jersey, 2003.

[Tagi09] R. Tagiew. "On Multi-Agent Petri Net Models for Computing Extensive Finite Games". In: N. Nguyen, R. P. Katarzyniak, and A. Janiak, Eds., *New Challenges in Compu Collective Intelligence - Studies in Computational Intelligence*, pp. 243–254, Springer, Berlin, 2009.

[Turo02] T. L. Turocy and B. von Stengel. "Game Theory". In: *Encyclopedia of Information Systems, Vol. 2*, pp. 403–420, Elsevier Science, 2002.

[Volc90] U. Völckers. "Arrival Planning and Sequencing with COMPAS-OP at the Frankfurt ATC-Center". In: *AACC/AIAA - American Control Conference*, San Diego, CA, 1990.

[Wasl06] S. L. Waslander, R. L. Raffard, and C. Tomlin. "Toward Efficient and Equitable Distributed Air Traffic Flow Control". In: *American Control Conference*, pp. 5189–5194, Minneapolis, Minnesota, USA, 2006.

[Wert05] B. Werther. *Kognitive Modellierung mit Farbigen Petrinetzen zur Analyse menschlichen Verhaltens*. PhD thesis, Technische Universität Braunschweig, Inst. f. Verkehrssicherheit u. Automatisierungstechnik, 2005.

[Wert06] B. Werther. "Colored Petri net based modeling of airport control processes". In: *Sydney International Conference on Computational Intelligence for Modelling, Control and Automation CIMCA06*, Sydney, Australia, 2006.

[Wert07] B. Werther, C. Möhlenbrink, and M. Rudolph. "Coloured Petri Net based Formal Airport Control Model for simulation and analysis of airport control processes". In: *Human Computer Interaction 2007: Human Interface and the Management of Information*, pp. 1027–1036, Springer Verlag, Beijing, China, 2007.

[West06] M. Westergaard. "Game Coloured Petri Nets". In: *7th Workshop on Coloured Petri Nets*, pp. 281–300, Aarhus, Denmark, 2006.

[West07] M. Westergaard, L. M. Kristensen, G. Brodal, and L. Arge. "The CombBack Method - Extending Hash Compaction with Backtracking". *28th International Conference on Application and Theory of Petri Nets and Other Models of Concurrency* -, Vol. 4546 of Springer Lecture Notes in Computer Science, pp. 445–464, 2007.

[Wick03] C. D. Wickens, J. Lee, D. Liu, Yili, and S. Gordon-Becker. *Introduction to Human Factors Engineering.* Prentice-Hall, Inc., 2003. 2nd edition.

[Zhou09] M. Zhou and N. Wu. *System Modeling and Control with Resource-Oriented Petri Nets.* CRC Press, 2009.

Im Rahmen von Forschungsarbeiten und Projektarbeiten im Lehrstuhl SRS und beim Deutschen Zentrum für Luft- und Raumfahrt (DLR) wurden von Dipl.-Ing. Hendrik Oberheid und Univ.-Prof. Dr.-Ing. Dirk Söffker die nachstehenden Diplomarbeiten betreut, wobei Bestandteile und Ergebnisse aus den Forschungs- und Projektarbeiten sowie den studentischen Qualifikationsarbeiten wechselseitig in die jeweiligen Arbeiten und somit auch in diese Promotionsarbeit eingeflossen sind.

[Gamr06] D. Gamrad. *Entwicklung von Mustern höherer Petri Netze zur rechnergestützten Simulation und Analyse von Situations-Operator-Modellen.* Diploma thesis, University of Duisburg-Essen, Chair of Dynamics and Control, 2005.

[Hass09] A. Hasselberg. *Entwicklung eines Fluglotsenassistenzsystems und zustandsraumbasierte Analyse des Bedienerverhaltens.* Diploma thesis, University of Duisburg-Essen, Chair of Dynamics and Control, 2009.

A. Appendix

A.1. Non-Standard Colourset Definitions

Listing A.1: Non-Standard Colourset Definitions

```
1  colset INT_list=list INT;
2  colset Callsign:STRING; (*Aircraft Callsign*)
3  colset ETA:INT;         (*Earliest Time of Arrival*)
4  colset LTA:INT;         (*Latest Time of Arrival*)
5  colset TTA:INT;         (*Target Time of Arrival*)
6  colset POS:INT;         (*Sequence Position*)
7  colset Aircraft=record cs:Callsign*eta:ETA*lta:LTA*tta:TTA*pos:POS
       *simcode:INT;
8  colset AmanInpList=list Aircraft; (*Traffic Situation Input*)
9  colset Q=product INT*INT_list;
10 colset AmanInpListXIntXINTList=product AmanInpList*Q; (*Planned
       Traffic Situation Output and Quality values*)
11 colset ETAchange=product Callsign*ETA;  (*Definition of EtaShift*)
12 colset ETAchangeList=list ETAchange;    (*List of EtaShifts*)
```

A.2. Rating Functions

Optimization function - EarlyETA

The function *EarlyETA*

$$EarlyETA \quad = \quad \frac{1}{n}\sum_{p=1}^{n}\sqrt{max\left(0,\left(\frac{t_{lim}-(tta_p-eta_p)}{t_{lim}}\right)\right)} \tag{A.1}$$

rates how close the assigned TTA of each aircraft comes to the submitted ETA of that aircraft. The closer these values of *tta* and *eta* are the higher the contribution of that aircraft to the quality value *EarlyETA*. The following elements enter the equation:

$$
\begin{aligned}
eta_p &= \text{earliest time of arrival of aircraft in position } p \\
tta_p &= \text{target time of of arrival of aircraft in position } p \\
t_{lim} &= 900\ s \text{ (delay treshold)} \\
n &= \text{number of aircraft in sequence} \\
p &= \text{sequence position}
\end{aligned}
$$

The parameter t_{lim} defines a delay treshold beyond which the individual quality contribution will be zero. Taking the square root favors many small delays over fewer big delays. It therefore supports a distribution of delays over different aircraft. The individual quality contributions are summed up and sum is devided by the number of aircraft n in the sequence, returning values between 0 and 1 as the result for $Q_{EarlyEta}$[1].

Stability function - SF_1

The function SF_1

$$
SF_1 = \frac{\sum_{ac=1}^{n} (ttaStab_{ac} \cdot wRel_{ac,k})}{\sum_{ac=1}^{n} wRel_{ac,k}} \tag{A.2}
$$

rates the stability of the sequence in cylce k by evaluating potential differences in aircraft target times with regard to cycle $k-1$.

The stability $ttaStab_{ac}$ for an individual aircraft ac is defined by a case distinction as

$$
ttaStab_{ac} = \begin{cases} 1, & dtta_{ac} < 0\,s \\ \left(1 - \frac{dtta_{ac}}{t_{sep}}\right), & (dtta_{ac} \geqslant 0\,s) \cap (dtta_{ac} \leqslant t_{sep}) \\ 0, & dtta_{ac} > t_{sep} \end{cases} \tag{A.3}
$$

where

[1] Note that in the CPN model implementation all quality values discussed in this section are realized as integers and are multiplied by the factor 10^5

$$
\begin{aligned}
dtta_{ac} &= tta_{ac,k} - tta_{ac,k-1} \\
t_{sep} &= 75\,s \text{ (minimum Separation)} \\
n &= \text{number of aircraft in sequence} \\
ac &= \text{aircraft ID.}
\end{aligned}
$$

By this definition, only positive values of *dtta* (delays) are interpreted as instabilities, negative *dtta* values (expeditions) are not penalized (case 1). For *dtta* values in the range of case 2, influence of delays $dtta_{ac}$ on quality is linear. Delays greater than t_{sep} lead to a stability values of zero for aircraft ac (case 3).

For the computation of SF_1 in equation A.2 potential TTA-shifts for aircraft with low sequence numbers are weighted higher (i.e. penalized harder) than instabilities for higher sequence numbers by multiplication with the factor $wRel_{ac,k}$

$$
wRel_{ac,k} = \left(1 - \frac{(pos_{ac,k} - 1)}{dg}\right) \tag{A.4}
$$

where the individual elements are defined as

$$
\begin{aligned}
pos_{ac,k} &= \text{position of aircraft } ac \text{ in cycle } k \\
dg &= 20 \quad \text{(degression range).}
\end{aligned}
$$

The paramter dg influences how quickly the influence of $ttaStab_{ac}$ decreases for aircraft with higher sequence numbers. The value of dg must be higher than the total number of aircraft n in the sequence.

The final quality value SF_1 is 1.0 if all aircrafts' TTAs remain stable or lie ealier than in the previous planning cycle. If one or more aircraft are delayed in terms of there TTAs, a value between 0.0 and 1.0 is returned.

Note that positional shifts have in this rating functions no direct impact on the result. However, shifts of sequence positions and target times are highly interrelated via the process itself and sequence shifts will almost inevitably delay one or more aircraft. This will affect SF_1.

Stability function - SF_2

The function SF_2

$$SF_2 = \frac{\sum_{p=1}^{n} (posStab_p) * wRel_p}{\sum_{p=1}^{n} wRel_p} \tag{A.5}$$

provides an alternative approach to rating sequence stability. It evaluates how many positions of the sequence in cylce k are still inhabited by the same aircraft as in cycle $k-1$.

The stability value $posStab_p$ for an individual sequence position p is defined by a case distinction as

$$posStab_p = \begin{cases} 1, & cs_{p,k} = cs_{p,k-1} \\ 0, & cs_{p,k} \neq cs_{p,k-1} \end{cases} \tag{A.6}$$

where

$$cs_{p,k} = \text{aircraft in sequence position } p \text{ at cycle } k.$$

Again, by introducing a factor $wRel_p$ in equation A.5

$$wRel_p = \left(1 - \frac{(p-1)}{dg}\right) \tag{A.7}$$

potential position shifts at low sequence numbers are weighted higher (i.e. penalized harder) than instabilities for higher sequence numbers. In the same way as for SF_2,the parameter dg (degression range) influences how quickly the weighting decreases for higher sequence numbersv ($dg > n$ must hold).

The final quality value SF_2 is 1.0 if all positions are inhabited by the same aircraft as in the preceeding cycle. If sequence switches have occured, a value between 0.0 and 1.0 is returned. The rating is not directly impacted by shifting target times. But again, target times and positions are closely linked via the process itself.

Stability function - SF_3

The rating function SF_3

$$SF_3 = \frac{(maxChange - actualChange)}{maxChange} \tag{A.8}$$

provides a third alternative to rate the stability of the sequence. It evaluates the magnitude of sequence changes (in contrast to SF_2 which analyzed their number).

In equation A.8 the variable *actualChange* of changes actually occured is related to *maxChange*, representing the maximimum changes possible for the given sequence length. The variable *actualChange* is defined as

$$actualChange = \sum_{ac=1}^{n} actPosDiff_{ac,k} * wRel_{ac,k} \tag{A.9}$$

with

$$\begin{aligned} actPosDiff_{ac,k} &= pos_{ac,k} - pos_{ac,k-1} \\ pos_{ac,k} &= \text{position of aicraft } ac \text{ in cycle } k. \end{aligned}$$

Further

$$maxChange = \sum_{ac=1}^{n} revPosDiff_{ac,k} * wRel_{ac,k} \tag{A.10}$$

with

$$\begin{aligned} revPosDiff_{ac,k} &= pos_{ac,k} - revpos_{ac,k} \\ revpos_{ac,k} &= (n - pos_{ac,k}) + 1. \end{aligned}$$

calculates a theoretical maximum of changes by reverting the whole sequence. In both *actualChange* and *maxChange* the switches for low sequence positions are penalized harder than for high sequence positions by multiplying with

$$wRel_{ac,k} = \left(1 - \frac{(pos_{ac,k} - 1)}{dg}\right). \tag{A.11}$$

Here, the following variable definitions

$$\begin{aligned} dg &= \text{degression range} \\ n &= \text{number of aircraft in sequence} \\ pos_{ac,k} &= \text{position of aicraft } ac \text{ in cycle } k \\ ac &= \text{aircraft} \end{aligned}$$

apply.

The final quality value SF_3 is 1.0 if all positions are still inhabited by the same aircraft and 0.0 if the sequence is reversed. If some sequence switches have occured, a value between 0.0 and 1.0 is returned. Changes over several positions are penalized harder than changes over just one position.

In contrast to stability function SF_2 which rated only the number of changs, SF_3 takes the magnitude of changes into account.

Total quality - $\mathbf{Q_{Total}}$

The function Q_{Total}

$$\begin{aligned} Q_{Total} &= (w_e \cdot EarlyETA) + (w_{SF_1} \cdot SF_1) + (w_{SF_2} \cdot SF_2) \\ &\quad + (w_{SF_3} \cdot SF_3) \end{aligned} \tag{A.12}$$

calculates a single quality value as the weighted sum of the four introduced rating functions $EarlyETA, SF_1, SF_2$, and SF_3. The corresponding weights w_e, w_{SF_1}, w_{SF_2} and w_{SF_3} are important parameters of the mechanism. They fundamentally determine the planning systems characteristics. Their variation span up the design space DS of mechanisms in chapters 6 and 7.

A.3. Scenario Generation

Characteristics of scenario set SC

In Figure A.1 to A.2, the characteristics of the resulting scenario set SC are represented in a graphical way.

Initial ETAs In Figure A.1, the distribution of earliest times of arrival ETAs for each aircraft A,B,C,D and for all aircraft together is shown. Each subplot shows a histogramm with number of occurences in six time ranges, the mean and the median ETA for the concerned aircraft.

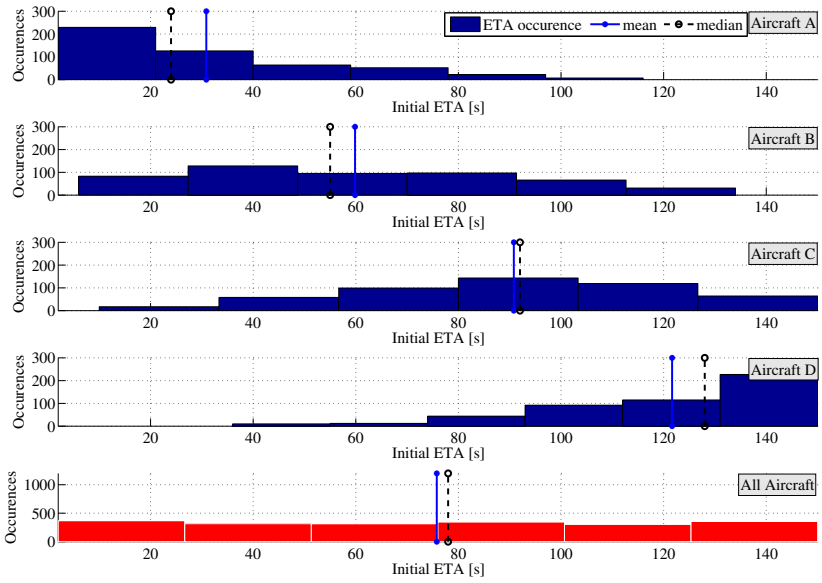

Figure A.1.: Distribution of initial ETAs over interval $[ET_{lower}, ET_{upper}]$for aircraft A,B,C, D (blue) and for all aircraft together (red).

Note that while time values tv_i for the four aircraft were drawn from a discrete uniform distribution (see MonteCarlo process, step 1), the convention to assign ~~tv~~tv values to aircraft A,B,C and D in ascending order makes that the resulting distributions for individual aircraft are nonuniform distributions. In particular, the order $mean(ETA_A) < mean(ETA_{B)} < mean(ETA_C) < mean(ETA_D)$ applies for the Scenario Set SC.

As the chosen range of ETAs with $ET_{upper} - ET_{lower} = 150$ s is only large enough for sequencing three of four aircraft with the required separation $t_{sep} = 75s$, a resource conflict is inherent in the scenarios, as some of the aircraft will experience delays through the presence of the others.

Initial TTAs In Figure A.2 the distribution of initial Target Times of Arrival TTA for the respective aircraft is shown as computed by step 3 of the MonteCarlo process. Again, each subplot shows a histogramm with number of occurences in the different time ranges, together with the mean and median TTA values.

Regarding the shape of the histogramm, this is nearly identical for all four aircraft, apart from a time shift to the right. This distribution is an effect of the resource conflict (too many aircraft on too short a time interval) in combination with the required time separation $t_{sep} = 75s$ which makes that the resulting sequence has maximum density and $TTA_i = TTA_{i-1} + t_{sep}$ applies with very few exceptions, so the same historgramm shape is mirrored with a shift of t_{sep} for all aircraft.

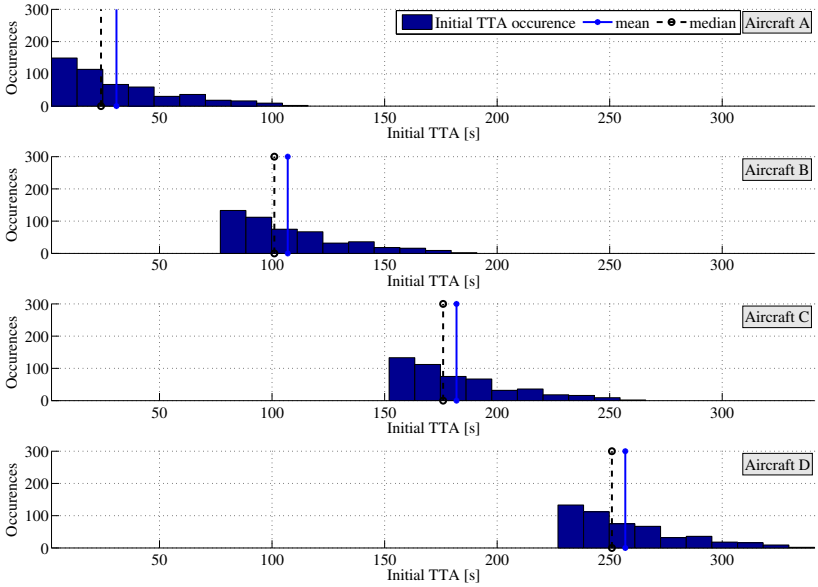

Figure A.2.: Distribution of initial Target Times of Arrival TTA for aircraft A,B,C and D

ETA-shifts In Figure A.3 the distribution of ETA-shifts for the four individual aircraft A,B,C,D and all aircraft together is shown. Each of the five suplots contains a histogram together with mean and median values for the ETA-shift. In this case, all shifts come from independent drawings of discrete uniform distributions. Deviations from this distribution are the effect of the bounded number of scenarios with $s = 500$ but are not systematic. The range of the ETAshifts lies in a time interval of $[-t_{sep,} + t_{sep}]$ that is approximately one time slot.

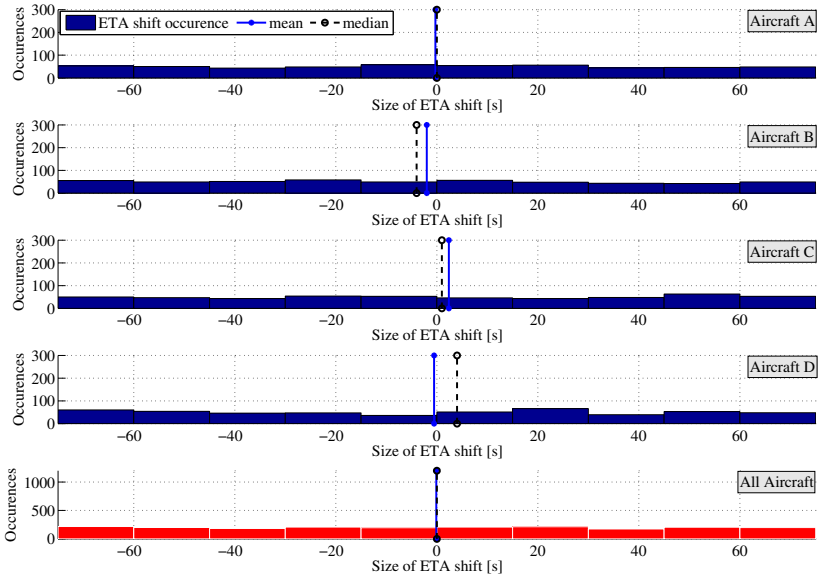

Figure A.3.: Distribution of ETA-shift sizes within interval $[ES_{lower}, ES_{upper}]$for aircraft A,B,C, D (blue) and all aircraft together (red)