Thesis for the degree of Doctor of Philosophy

Robust and Energy Efficient Scheduling

Nina Sundström

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2017

Robust and Energy Efficient Scheduling
Nina Sundström
ISBN 978-91-7597-634-1

© Nina Sundström, 2017.

Doktorsavhandlingar vid Chalmers tekniska högskola
Ny serie nr 4315
ISSN 0346-718X

Department of Electrical Engineering
Chalmers University of Technology
SE–412 96 Göteborg, Sweden
Telephone + 46 (0) 31 – 772 1000

Typeset by the author using LaTeX.

Printed by Chalmers Reproservice
Göteborg, Sweden 2017

to my beloveds Olle, Lou & Hjalmar

Abstract

Scheduling can generally be described as the act of allocating resources to tasks over time such that a given performance measure is optimized. Traditionally, disturbances are not considered while developing schedules. A rescheduling framework has thus emerged, aiming to minimize the impact of unforeseen disruptions. In this thesis, different rescheduling methods are proposed, where the obtained schedule is compared with the runtime schedule to evaluate the quality in terms of robustness and stability. The robustness is measured by the final time delay, whereas deviations in execution order or start time deviations act as stability measures.

A method is suggested, where the order in which tasks start in a time-optimal schedule is formulated as an event-based description. This results in a preserved execution order if delays are present. Logical restrictions are examined, and possibly relaxed, to avoid unnecessary delays. Another modeling approach presented, shows that an already established rescheduling method performed partly offline and partly online, called Affected Operations Rescheduling, can be performed completely offline.

From the aforementioned methods, stable schedules subject to sequence deviation are obtained. To generate both stable and robust schedules, a strategy is studied where idle time, so called slack, is inserted to schedules with the intention to absorb possible delays. Schedules are consequently protected against both start time deviations and makespan delay. In the literature on energy optimization, slack is diminished and often eliminated on behalf of reduced accelerations and velocities for robots, resulting in reduced energy consumption as well as extended execution times. The conflict between slack-based rescheduling techniques and energy optimization is highlighted in this thesis. The trade-off is evaluated by posing an optimization problem with measures of energy consumption, robustness and stability as criteria.

A challenge in production systems nowadays, especially with the increase in automation, is multiple resources sharing a restricted space. Much effort has been devoted to handle collision avoidance in such systems. A method is proposed in this work, where robustness is incorporated into trajectory planning for robots. When disruptions are present, the time at which a common workspace is expected to become available can differ. In the suggested approach, a clearance point is introduced, where the availability of the common workspace is evaluated. If the robot reaches this point and the shared space is not yet available, the robot has to be able to stop outside the shared space to avoid potential collisions. This requirement restricts the velocity at the clearance point. The impact on final time and energy consumption with respect to the position and timing related to the clearance point is studied. Results show the optimal clearance point position for different amount of slack available.

Keywords: Robust scheduling, energy optimization, trajectory planning, common workspace, discrete event systems.

Acknowledgments

What does it take to pursue a PhD? This question never entered my mind upon starting this journey about seven years ago. I was more in shock for given this opportunity, asking myself what I had gotten myself into. Just like the time when I, two months after enrolling to Engineering Physics, asked myself what I was thinking, having been away from school for six years. Well, to return to the initial question, I think I have the answer. Not surprisingly does it take hard work and dedication, but also one or numerous pinches of 'jävlar anamma' and in my case, great female role models to carry me through. Actually, when in doubts concerning the choice to continue with the physics studies my, at the time, almost 90-year-old grandmother said "failure is not an option, persist and you will eventually succeed". At first, the message felt harsh, but after earning a Bachelor's degree, Master's degree and hopefully a PhD degree, I am willing to say that she was right. I feel extremely fortunate to have met so many wonderful, inspiring and kind people on this journey, whom I would like to express my sincerest thanks to.

First, I would like to express my gratitude to my supervisor, Professor Bengt Lennartson, for your guidance when I couldn't find my way in the depths of the academic jungle, your valuable input and direction, leading me towards clarity, after which your encouragement to keep going has helped me to complete this work.

I would like to dedicate a special thanks to Lars, you are never forgotten. One time you said "once you get to know the postgraduate students, they are done with their PhD and leave." Tragically, you were the one to leave and never come back. Maybe you have hitchhiked to the Galaxy, to find out that the answer to life, universe and everything is not 42. Thank you for your friendship, you are forever missed.

I would like to express my appreciation to the team of administrators for being so very helpful. Thanks also to my colleagues at Systems and Control for your expertise in employeeship! Many thanks to Johan W, Johan N, Jonathan and Emilia for your company during lunch times and coffee breaks which always brings a smile to my face. Thank you Anna-Maria, Maliheh, Linnea, Mona, Sahar and Sathya for all the great conversations on both research but also motherhood and everything in between. Sarmad, thank you for your extreme kindness which has manifested itself in unlimited supply of lavashak over the years!

Oskar, my brother from another mother, you have walked alongside me during this journey and helped me in so many ways. Thanks for accepting the kind request to become my co-supervisor. Thanks also to my Chalmers family where, besides Oskar, Malin and Tomas are members. I miss the old times at Chalmers with you.

Furthermore, I am so grateful to my family for all your support over the years. I would also like to express my appreciation of my extended family and thank you for always helping out whenever needed. Finally, I would like to take the opportunity to thank you, Olle, for being so loving, understanding and patient throughout these years. You are my rock! I'm so grateful that you came into my life, and for the lives that you and I have created. Lou and Hjalmar, you mean the world to me, now there will be plenty of time available to play. I love you.

Nina Sundström
Göteborg, November 2017

List of Publications

This thesis is based on the following appended papers:

Paper 1. Nina Sundström, Oskar Wigström, Petter Falkman and Bengt Lennartson. "Optimization of Operation Sequences using Constraint Programming." *Proceedings of the 14th IFAC Symposium on Information Control Problems in Manufacturing, 45 (6), 1580-1585, 2012.*

Paper 2. Nina Sundström and Bengt Lennartson. "Event- and Time-Based Design of Operation Sequences with Uncertainties in Execution Times." *Proceedings of the IEEE 18th Conference on Emerging Technologies Factory Automation (ETFA), 2013.*

Paper 3. Nina Sundström and Bengt Lennartson. "Rescheduling Affected Operations - a Purely Predictive Approach." *Proceedings of the 13th International Workshop on Discrete Event Systems (WODES), 71-78, 2016.*

Paper 4. Nina Sundström, Oskar Wigström and Bengt Lennartson. "Conflict Between Energy, Stability, and Robustness in Production Schedules." *IEEE Transactions on Automation Science and Engineering, 14 (2), 658-668, 2017.*

Paper 5. Nina Sundström, Oskar Wigström and Bengt Lennartson. "Robust and Energy Efficient Trajectories in a Common Workspace Setting." *Submitted for possible journal publication.*

Other contributions by the author not included in this thesis, either due to content overlap with appended papers, or due to content outside the scope of this thesis:

Nina Sundström, Oskar Wigström, Sarmad Riazi and Bengt Lennartson. "On the Conflict Between Energy, Stability and Robustness in Production Schedules." *Proceedings of the IEEE International Conference on Automation Science and Engineering (CASE), 1263-1269, 2016.*

Nina Sundström and Bengt Lennartson. "From Time-Optimal Schedule to Robust Event-Based Control." *Proceedings of the IEEE 20th International Conference on Emerging Technologies and Factory Automation (ETFA), 2015.*

Oskar Wigström, **Nina Sundström** and Bengt Lennartson. "Optimization of Hybrid Systems with Known Paths." *Proceedings of the 4th IFAC Conference on Analysis and Design of Hybrid Systems, 45 (9), 39-45, 2012.*

Patrik Magnusson, **Nina Sundström**, Kristofer Bengtsson, Bengt Lennartson, Petter Falkman and Martin Fabian. "Planning Transport Sequences for Flexible Manufacturing Systems." *Proceedings of the 18th IFAC World Congress, 44 (1), 9494-9499, 2011.*

List of Acronyms

AOR	–	Affected Operations Rescheduling
CP	–	Constraint Programming
CSP	–	Constraint Satisfaction Problem
EFA	–	Extended Finite Automaton
FJSP	–	Flexible Job Shop Problem
JSP	–	Job Shop Problem
MINLP	–	Mixed Integer Nonlinear Program
NLP	–	Nonlinear Programming
RSR	–	Right-Shift Rescheduling
SP	–	Sequence Planner

Contents

Part I

Introductory Chapters

Chapter 1

Introduction

Subconsciously or not, most of us schedule our day time-efficiently when e.g. deciding who's taking the kids to school, dropping them off at soccer practice, what route to take to work, when to work-out or go shopping for groceries etc. Generally, the information required to develop a schedule is i) the tasks to perform, ii) the duration of each task, iii) precedence relations between tasks, e.g. taking the kids to school before going to work and iv) tasks that cannot be performed simultaneously, such as working out and grocery shopping. To exercise while going to work might on the other hand be feasible by e.g. taking the bike instead of the car. Also, alternatives can exist such as different routes to work, which can vary in duration.

Similar to our daily schedules, production systems are scheduled with the intention to optimize one or several criteria. The tasks can be to e.g. weld, glue, assemble, measure etc. The steps required to manufacture a product can be interpreted as precedence relations, e.g. parts should be glued before assembled. Furthermore, tasks processed by the same resource cannot execute simultaneously, if this would exceed the maximum capacity of the resource. Two robots sharing a common workspace are also prohibited to execute in parallel to avoid collisions. The execution order in which tasks are performed is thus constrained and the challenge lies in obtaining a schedule, where the resource conflicts are solved such that a criterion is optimized.

In this thesis, the development of robust and energy-efficient schedules is emphasized. Due to disruptions such as deviations in execution time or machine breakdowns, tasks can be delayed, which can affect the whole production system. To reduce the impact of disturbances, different approaches to develop robust schedules are proposed. The energy consumption for a robot is correlated with its acceleration. By reducing the velocity and acceleration, implying an extended execution time, energy can be saved. The duration for tasks can thus be decided such that energy consumption is minimized.

The work in this thesis began in 2010 as a part of a project, FLEXA, aiming to create tools and methods needed to automate the manufacturing of an aero engine structure. Due to low volumes and high cost for introducing automation, flexibility was deemed important, both in terms of products and parallel resources. Different modeling approaches were analyzed with the objective to minimize the time to manufacture 13 parts, each assembled by a set of smaller parts. Later, the thesis

work continued in connection to another project called AREUS, which focused on eco-design, eco-programming and Life Cycle Assessment of robotized factories. More specifically, the work in this thesis considers the generation of energy efficient time minimal robot trajectories.

The work in this thesis is also a result of an ongoing collaboration between Volvo Cars and the Automation Research Group at Chalmers, where the goal is to improve final time, energy consumption and robustness. The production environment at Volvo Cars features production lines with multiple robots sharing workspaces. Although the system has low flexibility and high throughput, it can be modeled as a problem within the area of job shop problems (JSPs), typically characterized by low volumes per product and high flexibility. Due to the extensive studies performed on JSPs in scheduling literature, this thesis often regards its problems from the viewpoint of JSPs.

1.1 Background

For more than a century, scheduling has been used in manufacturing systems in order to allocate resources to tasks over time. The need for a graphical description originated in production, where the workloads for workers and machines had to be planned (Gantt 1903). As a solution, Henry Gantt created the Gantt charts, which is still a popular tool for visualizing the timing of tasks. The optimization of schedules, i.e. the allocation of resources such that a given performance measure is optimized, has been extensively studied with publications appearing as early as in the 1950s (Johnson 1954). Areas of application are e.g. production, project management, computer science, health care and transportation (Silver et al. 1998; Kerzner 2013; Kwok and Ahmad 1999; An et al. 2012; Papadakos 2009).

Scheduling of production systems is the main focus in this thesis. Generally, a production system is divided into a set of *jobs* and a set of *resources* (Pinedo 2005). A job is usually reduced to a set of tasks with a given execution order. A resource refers to e.g. a machine, robot, tool, worker etc. In literature, the terms task and *operation* are commonly interchangeable for production systems. The goal, called objective, can e.g. be to minimize the completion time of the system, referred to as the makespan. For some systems, jobs have due dates and in this case it is common to e.g. minimize the lateness of jobs, which describes if jobs end early or late. If only late jobs should be penalized, this is referred to as minimizing the tardiness.

Traditionally, deterministic systems are considered in scheduling (Graham et al. 1979; Taillard 1993). However, production systems are subject to a number of different disruptions, e.g.

- Process time variation
- Machine breakdown
- Machine maintenance
- Tool breakdown/Tool tear

Due to the stochastic nature of production systems, a framework called rescheduling has emerged, aiming at developing schedules that are insensitive to unforeseen disruptions (Vieira et al. 2003). Numerous techniques have been proposed to cope with disturbances (Sabuncuoglu and Goren 2009). In redundancy-based methods, additional idle time is inserted into the schedule, with the purpose of absorbing possible delays (Jorge Leon et al. 1994; Lambrechts et al. 2011; Davenport et al. 2014). Another approach focuses on maintaining the execution order in the schedule when disturbances are present. The schedules developed in these approaches are referred to as baseline schedules.

To evaluate the quality of baseline schedules, concerning the ability to withstand disruptions, measures of *robustness* and *stability* are addressed in rescheduling (Goren and Sabuncuoglu 2008). For a robust schedule, the performance does not deteriorate in the presence of disturbances. The makespan deviation between the runtime schedule and the baseline schedule is commonly used as a robustness measure. A schedule is said to be stable if the realization of the schedule does not deviate from the baseline schedule. The stability is frequently measured by the start time deviation, i.e. the start time for operations in the runtime schedule is compared with the start time for the corresponding operations in the baseline schedule. Another type of stability measure is the deviation in execution order between the baseline and realized schedule (Sabuncuoglu and Goren 2009; Katragjini et al. 2013).

Energy minimization is increasingly used as an objective in scheduling (Dietmair and Verl 2009; Dai et al. 2013). In (Vergnano et al. 2012), a method to reduce the energy consumption of robots is proposed by reducing, and often eliminating, idle time between operations. The energy consumption of robots is embedded into the scheduling model, where each operation has an energy consumption signature, parametrized by its execution time. This signature is also considered in (Riazi et al. 2017), where the results show that energy can be saved, even without changing the operation sequences. As a result of common workspaces, the schedule contains gaps, i.e. idle time, where a robot waits for the shared space to become available. By reducing the velocity and acceleration for a waiting robot, and thus extending the execution time such that the idle time diminishes, a reduction in energy consumption is achieved without affecting the final time.

In (Salido et al. 2015), energy consumption and robustness of schedules are studied, where machines with three modes is regarded. Each mode corresponds to a certain processing time and energy consumption. A shorter processing time is associated with higher energy consumption. Hence, a decrease in energy consumption implies a longer makespan. Robustness is measured by the ability to recover from a delay, by running the machine at a higher speed. The slack is thus diminished on behalf of longer processing times, which is advantageous both in terms of energy consumption and robustness. This approach is feasible due to the multi-modes machines. However, robustness was not included into the optimization model, only estimated by simulating random delays in a schedule generated with makespan and energy consumption as criteria.

Due to the increase in automation and limited space, robots usually work closely together. Collision avoidance is hence an important topic in multi-robot systems

(Liu et al. 2000). Apart from assigning resources to operations, scheduling is also used to decide the order in which robots can use a common workspace. Disjunctive constraints are frequently used to model collision avoidance. Typically, multi-robot systems are coordinated without considering robot dynamics (LaValle and Hutchinson 1998; Siméon et al. 2002), while work related to trajectory planning for a single robot does. In (Bobrow et al. 1985), a space formulation for a single robot generates a time-optimal trajectory for a specified path, while regarding dynamics. A method to integrate robot dynamics into a multi-robot system model is presented in (Peng and Akella 2005). A time dependent formulation is posed, where the robots are modeled as double integrators and the path is assumed to be known. In (Wigström et al. 2017), computationally efficient space formulations for the multi-robot system, with multiple objectives including final time and energy consumption, are presented. Prespecified paths as well as double integrator models for robots are considered. None of the work on trajectory planning has included robustness in terms of uncertainties concerning the time at which the common workspace becomes available.

1.2 Research Questions

Based on the background previously described, different areas arise where further research is required. To set the boundaries of the topics to explore, the following research questions are to be answered:

RQ1 Based on a general timed discrete event system, how can the system design and optimization be integrated? How can the optimization results be formalized and retrieved by the original model?

RQ2 How can disturbances be accounted for?

RQ3 How can the potential conflict, which appears when energy optimization is combined with redundancy-based rescheduling techniques, be evaluated and what is the trade-off?

RQ4 Regarding collision avoidance, how can robustness be included in a problem formulation? How does the incorporation of robustness affect the performance?

1.3 Contributions

A general theme of this thesis concerns how to handle uncertainties in production systems. The research areas considered are: i) modeling and scheduling of production systems, including protection against disruptions, ii) energy optimization versus rescheduling and iii) robust trajectory planning. Attempting to answer research questions RQ1-RQ4, resulted in the following contributions.

C1 An operation model representing a general shop floor, modeled in Sequence Planner (SP), is formulated as a constraint programming (CP) model to

develop a minimum time schedule. An abstraction method, work equivalence, is proposed resulting in computational benefits. To close the loop, the conditions in the original operation model are altered, to also reflect the execution order in the time minimum schedule. The conditions are evaluated and, if possible, relaxed to avoid unnecessary delays if uncertainties are present. Due to the event-based description of the schedule, the related timing of operations, such as start times, can be disregarded.

In Paper 1, the integrated design and optimization of a system is presented. The contribution in Paper 2 concerns the remaining part of the contribution described above. Paper 1 and Paper 2 answer RQ1. Since the timing can be ignored as a consequence of the event-based formulation, the contribution in Paper 2 also answers RQ2.

C2 For a special case of scheduling problem, the JSP, a rescheduling method called Affected Operations Rescheduling (AOR) has been proposed. An initial time-optimal schedule is first developed and, if disruptions are present, the operations are rescheduled online. Only operations directly or indirectly affected by disturbances are delayed. In scheduling, disjunctive graphs are commonly used to model and optimize JSPs. An offline approach is proposed by using the disjunctive graph, resulting in the same behavior as AOR. A conservative approach, Right-Shift Rescheduling (RSR), commonly used as a benchmark in the rescheduling literature, delays all succeeding operations if disruptions are present. A proof is posed to show that the performance of AOR is always better than, or equal to, RSR.

The contribution is presented in Paper 3 and answers both RQ1 and RQ2 when a JSP is considered. If disturbances are present, the concept in Paper 3 is less restrictive, compared to the approach in Paper 2.

C3 A systematic method is proposed to evaluate the conflict between energy consumption, robustness and stability. An energy consumption signature, parameterized by processing time, is derived based on experiments on an industrial robot. The accuracy of existing surrogate measures for robustness and stability is analyzed together with a proposed stability measure. To study the trade-off, the best performing measures, together with makespan and energy consumption, are used in a multi-objective mixed-integer quadratically constrained optimization formulation. The suggested stability measure enables for different operation sequences to be examined. The results show that a conflict between energy efficiency, robustness and stability exists. A decrease in energy consumption yields a decrease in robustness and stability. Also, an increase in robustness results in a decrease in stability and vice versa.

The contribution is presented in Paper 4 and answers RQ2 and RQ3.

C4 To include robustness into trajectory planning where common workspaces exist, a space formulation is posed including robot dynamics with final time and energy consumption as criteria. A predefined path is assumed. Robust

constraints are included as velocity and timing constraints regarding a clearance point. At this point, a decision is taken to either enter, or come to a halt at the boundary of a shared space. The effect on the performance is evaluated for different positions of the clearance point, as well as the timing corresponding to when the clearance point can be traversed. The analysis alleviates the decision concerning the location of the clearance point if disruptions in the time at which the shared space becomes available are present.

The contribution is presented in Paper 5 and answers RQ2 and RQ4.

1.4 Thesis Outline

The thesis is outlined as follows. Chapter 2 addresses scheduling for timed discrete event systems. Different problems, commonly considered in the scheduling literature, are presented. The tool SP which can be used to e.g. model operation sequences is introduced together with the SP model and its underlying extended finite automaton (EFA). The translation of the SP model to a CP model is discussed and retrieving the optimization result to SP is discussed. Furthermore, the disjunctive graph is presented.

The rescheduling framework is addressed in Chapter 3. Rescheduling strategies and underlying methods are presented, where two approaches, AOR and RSR, are explained in more detail. Schedule quality is discussed together with two terms used to assess the quality, robustness and stability. Measures of robustness and stability are stated in Chapter 4. Existing surrogate measures are presented as well as a proposed surrogate stability measure. These measures are individually included as objectives in an optimization formulation to study the computational efficiency and the ability to withstand disruptions for the resulting schedules.

Energy optimization is presented in Chapter 5, together with the energy consumption signature. A method to combine the results from Chapter 4 with an energy consumption measure is proposed in this chapter. Trade-off analysis is performed to evaluate the conflict.

A problem formulation, including robustness into trajectory planning is presented in Chapter 6, which is posed in space and assumes a predefined path. The objectives included are final time and energy consumption. Constraints concerning robustness are discussed as well as their affect on system performance. Finally, the appended papers are summarized in Chapter 7, after which conclusions and future work are discussed in Chapter 8.

Chapter 2

Scheduling for Timed Discrete Event Systems

In scheduling, resources are allocated to operations over time such that a given performance measure is optimized. The literature on scheduling is dense, see e.g. (Pinedo 2005)(Hooker 2005)(Herrmann 2006). In this chapter, commonly regarded scheduling problems are presented. The software SP is introduced, where precedence relations and resource requirements can be modeled (Lennartson et al. 2010). The SP model is presented as well as an approach to translate this model to a CP model. For JSPs, the disjunctive graph is frequently used for modeling and optimization. The disjunctive graph is thus presented together with its properties. The outcome of both approaches are minimum makespan schedules.

2.1 Scheduling Problems

Scheduling problems are characterized by the tasks to perform and their duration. Also, knowledge concerning precedence relations in each job and resource requirements are necessary. Based on the characteristics, the problems can be divided into subclasses describing e.g. the amount of flexibility. In flexible systems, an operation can be performed by a set of resources. For even more complex systems, an operation can also require several resources. In this section, a selection of common scheduling problems is presented.

Flow shop problem In the flow shop problem (FSP), there are m resources in sequence. The resources have unit capacity, referred to as unary resources. Each job has to be processed on all resources, i.e. the operations corresponding to a job are each processed on different resources. All jobs follow the same route, i.e. first processed on Resource 1, then on Resource 2 and so on. After an operation is completed on one resource, the next operation in the job is placed in a queue to the next resource. Usually, the queues are handled by the first in first out rule.

Job shop problem In a JSP, m unit capacity resources in sequence is also considered. However, the sequence in which the resources are used is not identical for

the different jobs. As for the FSP, the operations in a job are processed on individual resources. A special case of JSP is when operations can revisit resources, called recirculation.

Flexible job shop problem As the name suggests, the flexible job shop problem (FJSP) is an extension of the JSP. In this case, multiple resources can perform the same operation. A modeling approach for FJSP is proposed in Paper 1, where an abstraction method, called work equivalence, is presented. In this method, a set of c identical unary resources can be modeled as one resource with capacity c. Different modeling approaches are compared by analyzing the performance, see Paper 1 for more information.

Resource constrained project scheduling problem In the resource constrained project scheduling problem (RCPSP), an operation can require multiple resources, either of the same type or of different types (or both). This problem was originally considered for project scheduling, where the operations are activities and a common resource type is personnel, which explains the requirement to have multiple resources of the same type.

The first two papers appended in this thesis consider general systems, whereas the following two papers focus on JSPs. In the last paper, where robust trajectory planning is studied, no specific environment is presumed. To schedule the problems, SP is introduced as a means to model the systems, after which an approach to transform the SP model to an optimization model is proposed.

2.2 Sequence Planner

For more than a century, scheduling has been used in manufacturing systems in order to decide when operations are to be performed and by what resource. Gantt charts were early used to establish the timing of operations and is still today a popular tool for graphical representation of operation sequences (J. M. Wilson 2003). Other popular tools are e.g. Microsoft Excel and PERT charts (Levin and Kirkpatrick 1966)(Kerzner 2013). A more recent example is the modeling tool SP (Lennartson et al. 2010). Compared to Gantt charts, more complex operation sequences can be expressed in SP such as alternative sequences.

A major advantage of SP is that it can be used for both the development of the product design, process design and the related control system. Hence, the need for a tool like SP is highly motivated due to the possibility to integrate these areas. Throughout the development process; new requirements can easily be added, the system can be optimized, the result visualized, and control code generated. Hence, the process designer can, if necessary, revise the operation sequences and iterate the procedure until a desired system behavior is achieved.

In the following sections, the SP model is presented, as well as the contribution in Paper 1, where the SP model is converted to a CP model to obtain a schedule. Then, based on the resulting schedule, conditions are generated to maintain the execution

order in the schedule. Hence, the result from the CP model is retrieved by the SP model as additional conditions. This contribution is presented in Paper 2. Also, an approach to relax the conditions is proposed. In this case, unnecessary delays can be avoided in the presence of disruptions in the system.

2.2.1 Sequence Planner model

The SP model has an underlying logical operation model, given by an extended finite automaton (EFA). With this model, it is possible to use preconditions expressing when operations can start. First, we will introduce the EFA operation model as well as how resource booking is modeled. Then, the corresponding SP model is introduced.

Definition 2.2.1 (Operation Model)**.** An operation can be modeled as an extended finite automaton where the set of locations $Q_j = \{O_j^i, O_j^e, O_j^f\}$, the event set $\Sigma_j = \{O_j^\uparrow, O_j^\downarrow\}$, the set of transition conditions $C_j = \{C_j^\uparrow, C_j^\downarrow\}$, the transition relation $\rightarrow_j = \{\langle O_j^i, O_j^\uparrow / C_j^\uparrow, O_j^e\rangle, \langle O_j^e, O_j^\downarrow / C_j^\downarrow, O_j^f\rangle\}$ and the initial location $q_j^i = O_j^i$, see Fig. 2.1.

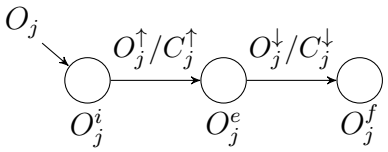


Figure 2.1: EFA for an operation O_j.

A transition between two locations is enabled when the transition condition is satisfied. The transition is then performed when the event occurs. For operation O_j, the transition from the initial location O_j^i to the executing location O_j^e is enabled when the precondition $C_j^\uparrow$ is satisfied. The transition is then performed when the start event $O_j^\uparrow$ occurs. Similarly, the completion event $O_j^\downarrow$ can only occur when the postcondition $C_j^\downarrow$ is satisfied.

The resource booking for an operation O_j can be specified in the pre- and postconditions, $C_j^\uparrow$ and $C_j^\downarrow$. Let $\mathcal{R}$ be the set of resources. For a unary resource, modeled by a variable $R \in \mathcal{R}$, a value $R = 0$ implies that the resource is available. The resource is booked by the next value condition, $\acute{R} = 1$, which defines the next value of R after the transition, at the same time as $O_j^\uparrow$ occurs. Hence, the precondition $C_j^\uparrow$ is given by

$$R^+ \equiv R = 0 \wedge \acute{R} = 1$$

where $\acute{R}$ is the next value of R. Similarly, the unbooking of a resource is given by

$$R^- \equiv R = 1 \wedge \acute{R} = 0$$

The SP model is a high-level representation of the EFA operation model. As previously mentioned, a production system can be split up into operations and

resources. The jobs are represented by operation sequences. In SP, the core of the operation model is the pre- and postconditions, i.e. $C_j^{\uparrow}$ and $C_j^{\downarrow}$ for an operation O_j. These conditions can be used to specify relations between operations. Also, the resource booking and releasing of a resource are included in the preconditions and postconditions.

A graphical representation of a straight sequence specifying an operation relation between two operations, O_{11} and O_{12}, is depicted in Fig. 2.2a. Resource booking and unbooking of two resources, R_1 and R_2, are included. The operations can also be viewed as self-contained by reformulating the preconditions to include operation relations. In Fig. 2.2b, the relation between operation O_{11} and O_{12} is expressed in the precondition $C_{12}^{\uparrow}$ for operation O_{12}. Hence, the pre- and postconditions for O_{11} and O_{12} are specified as

$$C_{11}^{\uparrow} = R_1^+ \qquad C_{11}^{\downarrow} = R_1^-$$
$$C_{12}^{\uparrow} = O_{11}^f \wedge R_2^+ \qquad C_{12}^{\downarrow} = R_2^-$$

(a) Operations in sequence (b) Self-contained operations

Figure 2.2: Different representations of operation relations.

In this section, modeling of operation sequences as well as resource booking in SP were presented. Next, the procedure to map this information to an optimization model is introduced in order to generate a production schedule.

2.2.2 Scheduling using SP and CP

CP has its roots in the artificial intelligence and computer science communities (Hooker 2000). Originally it was applied to constraint satisfaction problems. The constraints can be viewed as relations, and the result from the constraint satisfaction problem states what relations should hold among the decision variables (Rossi et al. 2006). In more recent decades, CP has evolved to also include optimization. In scheduling, this corresponds to assigning start times to operations such that the relations between operations are satisfied, as well as the resource constraints where the resource usage can not exceed its capacity. An objective can e.g. be to minimize the makespan, which corresponds to minimizing the maximum completion time for the last operation in each job sequence.

The operation relations as well as the resource booking is stated in the pre- and postconditions in the SP model. Hence, the relations between operations based on the job sequences can be expressed with constraints such as e.g. endBeforeStart() in IBM ILOG CP Optimizer. For the resource booking, a unary resource can be modeled with a disjunctive constraint such as noOverlap() in IBM ILOG CP Optimizer to specify that operations sharing a resource cannot execute simultaneously. For resources with

a capacity greater than one, a cumulative constraint can used to express that the sum of operations that utilizes the resource are bounded from above by the capacity of the resource. As a result from the optimization, a schedule including the operations and their corresponding start times are generated. The procedure to transform the operation model in SP to a CP model and generate a minimum makespan schedule is a part of the contribution in Paper 1.

2.2.3 Extending the SP model

The relations based on the execution order in the optimal schedule can be combined with the relations in the SP model. Once the relations are retrieved by SP as conditions, the resulting sequences can be visualized. Let $\mathcal{O}$ denote the set of operations and $\mathcal{N}$ the corresponding index set. For operation $O_j \in \mathcal{O}$, its index $j \in \mathcal{N}$. Let $\mathcal{P}_j$ denote the index set of preceding operations to O_j. In order to maintain the execution order in the optimal schedule, a restrictive approach is to forbid operations to start before all preceding operations have been completed. This can be achieved by

$$C_j^{\uparrow} = \bigwedge_{i \in \mathcal{P}_j} O_i^f, \quad j \in \mathcal{N},\ O_i \in \mathcal{O} \tag{2.1}$$

If delays are present, the conditions in (2.1) will guarantee that the execution order in the schedule is maintained. Most certainly, not all pairs of operations are related, i.e. belonging to the same job or executed by the same resource. Hence, a procedure to assess the relations will later be introduced with the aim to relax conditions such that unnecessary delays are avoided when an disruption occur.

For general shop floors with resource capacities strictly greater than one, forbidden sets are introduced to represent the sets of operations with resource conflicts, i.e. where the resource requirements exceed the resource capacity. A minimal forbidden set is a set such that each subset cannot contain a resource conflict. In a minimal forbidden set, a resource conflict can be solved by posting a single constraint between two competing operations. In a job shop environment, due to unary resources, all pairs of operations realized by the same resource are in conflict, hence the minimal forbidden set has size two. In this case, the resource conflicts can be solved by invoking an order, in which the operations should be performed. However, for more general systems, the number of minimal forbidden sets is exponential in the number of operations. Hence, many publications have focused on ways to overcome the limitation imposed by the large number of minimal forbidden sets (Lombardi et al. 2013; Lamas and Demeulemeester 2015).

For example, in (Cesta et al. 2015), the authors use a constraint-based scheduling approach for the resource constrained project scheduling problem. Their approach is based on the formulation of the problem as a constraint satisfaction problem. More specifically, the problem is first solved as a simple temporal problem network where the resource constraints are disregarded. Then, the authors propose an approach to study the resource usage over time, called resource profiles, to check if the resource requirements are greater than the capacities. If minimal forbidden sets are detected,

resource constraints are added to the model. Constraint propagation is performed in the temporal graph to check temporal consistency, resulting in a new solution. This is an iterative procedure which is performed until either the temporal graph becomes inconsistent or, the resource profiles are consistent with the resource capacities. In the latter case, the result is a simple temporal network with precedence constraints, not only belonging to the initial temporal problem, but also due to sequencing on the resources.

In Paper 2, a method to analyze the constraints in (2.1) is proposed. All relations not specified in the SP model are evaluated and possibly relaxed. Feasibility tests are performed to check if operations share a resource and if not, constraint propagation is performed to see if an operation can start before a prior operation is completed. However, the order in which the operations start is restrained, i.e. the preceding operation has to start before the succeeding operation starts. Thus, if a feasibility test is passed, conditions are generated expressing that the succeeding operation can start during, or after, the execution of the preceding operation. Consider a preceding operation $O_i,\ i \in \mathcal{P}_j$. If the feasibility test is passed, the constraint is updated to

$$C_j^{\uparrow} = O_i^e \vee O_i^f, \quad j \in \mathcal{N},\ i \in \mathcal{P}_j,\ O_i \in \mathcal{O} \tag{2.2}$$

However, if the test fails, the succeeding operation is constrained to start after the completion of the preceding operation, i.e.

$$C_j^{\uparrow} = O_i^f, \quad j \in \mathcal{N},\ i \in \mathcal{P}_j,\ O_i \in \mathcal{O} \tag{2.3}$$

The proposed method is restricted to consider at most two preceding operations in the schedule when performing the feasibility tests. However, when using minimal forbidden sets, the solution strategy is much more complex compared to the proposed method. For more information regarding this procedure, see Paper 2. Note that conditions should be added to match up with the initial execution order to guarantee feasible sequences. This can be achieved by adding a condition for all pairs of operations $O_i, i \in \mathcal{P}_j$ and O_j, where relaxation was viable resulting in condition (2.2). The condition

$$C_k^{\uparrow} = O_i^f \wedge C_k^{\uparrow}, \quad j \in \mathcal{P}_k,\ i \in \mathcal{P}_j,\ k \in \mathcal{N} \tag{2.4}$$

should be included, which adds O_i to the precondition for O_k. If O_i is delayed such that the completion time exceeds the time at which O_j is completed, any succeeding operation $O_k, j \in P_k$ is consequently prohibited to start before the completion of O_i. As a result, the behavior of the original schedule is eventually reestablished.

Recall that SP can be used to model general shop floors. For a job shop, the disjunctive graph is commonly used to model and optimize the system. Next, the disjunctive graph together with its properties are introduced.

2.3 Disjunctive Graph

Due to the property of a job shop where resources have unit capacity, the disjunctive graph is suitable for modeling and scheduling of such systems. A job shop problem

can be represented as a disjunctive graph $G(\mathcal{N}, \mathcal{A}, \mathcal{E})$, where $\mathcal{N} = \{1..n\}$ is the set of nodes corresponding to the operations to be processed on the set of resources $\mathcal{R}$, see Fig. 2.3. Node $i \in \mathcal{N}$ corresponds to operation O_i. Hence, the set of nodes and the index set previously presented are isomorphic. Two dummy nodes, U and V, represent a source and sink node. The set $\mathcal{A} \subseteq \mathcal{N}^2$ corresponds to pairwise precedence relations among operations. The precedence relations are represented by conjunctive arcs. An arc $(i, j) \in \mathcal{A}$ indicates that operation O_i has to be completed before operation O_j can start. The conjunctive arcs are depicted as solid arcs in Fig. 2.3.

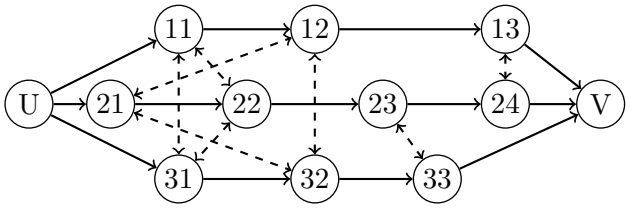


Figure 2.3: Disjunctive graph $G(\mathcal{N}, \mathcal{A}, \mathcal{E})$.

Due to unary resources, operations sharing a resource may not execute in parallel. The set $\mathcal{E}_R \subseteq \mathcal{N}^2$ corresponds to pairwise precedence relations between operations processed by a resource $R \in \mathcal{R}$. These relations are represented by disjunctive arcs. The set $\mathcal{E} = \cup_{R \in \mathcal{R}} \mathcal{E}_R$ is the total set of disjunctive arcs, depicted as dashed arcs in Fig. 2.3.

In order to find a feasible schedule, the disjunctive arcs have to be resolved, i.e. the pairwise precedence order on the resources has to be determined, such that the resulting graph is acyclic. The processing time for an operation is represented as a weight on the arcs emanating from the corresponding node. Arcs emanating from the source node has zero weight. A path between two nodes is represented by the sum of weights on arcs connecting the nodes. Among the set of feasible schedules, the time-optimal schedule has the shortest longest path from the source node to the sink node (Pinedo 2005). In the resolved graph, redundant arcs can be removed due to transitive relations. A transitive relation is given by

$$(a, b) \ \wedge \ (b, c) \ \Rightarrow \ (a, c), \qquad a, b, c \in \mathcal{N}$$

For a job shop, each node has at most two incoming and outgoing arcs respectively in the resulting non-transitive graph. Each incoming arc represents a relation to a preceding operation in, either the job, or the resource sequence. The outgoing arcs represent relations to succeeding operations in the job and resource sequences. The resulting resource sequences will be straight operation sequences as a consequence of unary resources. Parallel sequences would violate the bounds on the resource capacities with are equal to one. Hence, an operation has at most one preceding/succeeding operation in the job, as well as in the resource sequence.

The non-transitive acyclic graph in Fig. 2.4 represents a feasible schedule based on the disjunctive graph in Fig. 2.3, where the disjunctive arcs show that e.g. nodes 11, 22 and 31 share a resource. The resulting resource sequence for this specific

resource is given by 11, then 31 and finally 22 based on the arcs in Fig. 2.4. Note the straight job and resource sequences. The relations between operations can thus be retrieved by studying the direction of the arcs.

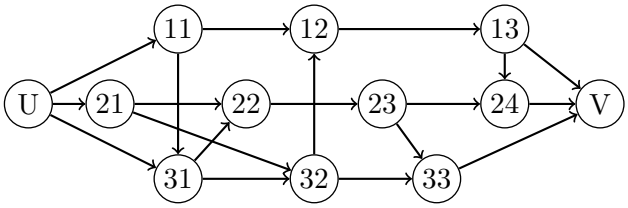


Figure 2.4: Directed acyclic graph representing a feasible solution to the disjunctive graph in Fig. 2.3. The disjunctive arcs have been resolved and the resulting graph is acyclic. Note that the number of arcs have been reduced due to transitive relations.

So far schedules have been generated either by using SP combined with CP or by using the disjunctive graph. Two terms used to analyze schedules are free slack and total slack. The free slack corresponds to the time that an operation can be delayed without delaying the start of its immediate successors.

Definition 2.3.1 (Free slack)**.** Let t_i denote the start of operation O_i while d_i denotes its duration. The set $\mathcal{S}_i$ constitutes the immediate successors to O_i. The free slack for operation O_i, denoted by s_i, is given by

$$s_i = \min_{j \in \mathcal{S}_i} t_j - t_i - d_i \tag{2.5}$$

The free slack can be viewed as a local property since it considers the impact on the immediate successor to an operation. The total slack describes how much an operation can be delayed without affecting the makespan of the system. Hence, the total slack is considered as a global property.

Definition 2.3.2 (Total slack)**.** Let $\ell_{i,j}$ denote the longest path between node i and node j. The latest allowable start of operation O_i without affecting the makespan is given by $T - \ell_{i,V}$ where V is the sink node and T represents the makespan. The total slack, also referred to as the total float, of O_i, denoted by f_i, is given by

$$f_i = T - \ell_{i,V} - t_i \tag{2.6}$$

In the following text, ℓ_i implies the longest path between node i and the sink node V. Free slack and total slack play a crucial role when defining measures related to robustness and stability, introduced in Chapter 4. First, this chapter is summarized after which rescheduling and some of its underlying methods are introduced.

2.4 Summary

Scheduling, together with different scheduling problems commonly considered, were presented in this chapter. Modeling of operation sequences in the tool SP was

introduced. A method proposed to transform the sequences of a general problem in SP to a CP model was discussed, as well as an approach to retrieve the resulting schedule behavior to the SP model. The disjunctive graph was presented, which is extensively used for systems with straight operation sequences and unit capacity resources, i.e. systems without flexibility.

Next, the rescheduling framework is presented, where uncertainties in the production systems are taken into consideration when schedules are developed.

Chapter 3

Rescheduling

Usually, static and deterministic environments are considered when developing production schedules. However, real-life production systems are subject to unexpected disruptions, not considered when generating the schedules. This includes e.g. machine breakdowns, over- and underestimation of processing time, job cancellations and machine repairs (Pinedo 2012). In response to these disruptions, rescheduling is used to minimize the impact on the schedule performance when disturbances are present (Vieira et al. 2003)(Sabuncuoglu and Goren 2009).

Several rescheduling methods have been proposed in the literature. Basically, the methods can be divided into three main strategies; proactive scheduling, predictive-reactive scheduling and reactive scheduling. In this chapter, the different strategies and their underlying methods are introduced. Also, stability and robustness are defined, describing how well a schedule withstands disruptions. Two rescheduling methods are studied more in-depth, Right-Shift Rescheduling (RSR) and Affected Operation Rescheduling (AOR).

3.1 Strategies and Methods

The classification of a rescheduling method depends on whether the method is performed offline, online or a combination of the two, see Fig. 3.1. As previously mentioned, the rescheduling strategies are; proactive scheduling, predictive-reactive scheduling and reactive scheduling. Next, the different strategies, and a selection of underlying methods, are presented.

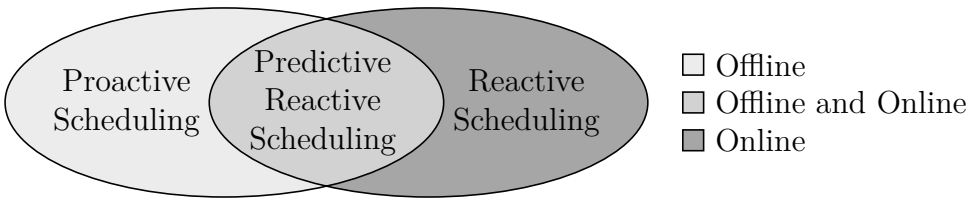


Figure 3.1: The different strategies considered in rescheduling.

3.1.1 Proactive Scheduling

In proactive scheduling, schedules are generated offline with embedded protection against disruptions. These schedules are referred to as baseline schedules. The underlying methods are in general based on redundancy techniques or on probabilistic techniques (Beck and N. Wilson 2007)(Lou et al. 2012). The methods based on probabilistic techniques include uncertainties with known probability density functions. In (Lamas and Demeulemeester 2015), uncertainties in processing times are considered. A robustness measure is proposed, based on the joint probability that an operation in the realized schedule will start at the same time as intended in the baseline schedule. Other methods, using optimization under uncertainty to develop baseline schedules, are presented in (Diwekar 2008)(Sahinidis 2004).

If knowledge regarding the disruptions is unavailable, redundancy-based methods are considered. In these methods, redundant time is added to the schedule in order to make it less sensitive to disruptions during execution (Van de Vonder et al. 2008; Daniels and Kouvelis 1995). In (Mehta and Uzsoy 1998), idle time is added to account for machine breakdowns, while time buffers are used to protect the schedule against processing time variability in (Van de Vonder et al. 2008). In (Jorge Leon et al. 1994), different robustness measures are evaluated by, for each measure, posing an objective including makespan and robustness measure, to analyze the system performance of baseline schedules including additional idle time. In (Lambrechts et al. 2008)(Hazır et al. 2010), more recent approaches, suggesting different robustness measures, are presented. The schedule run in real-time is referred to as the realized schedule. Hence, the main idea in proactive scheduling is for the realized schedule to follow the baseline schedule as closely as possible. In this thesis, Paper 4 presents a proactive approach to insert additional slack into a schedule.

3.1.2 Predictive-Reactive Scheduling

In predictive-reactive scheduling, an initial deterministic schedule is first generated offline. In response to disruptions, the existing schedule is updated online, during execution. The most conservative method, RSR, globally right-shifts all remaining operations when a disruption occurs (Raheja and Subramaniam 2002)(Wu et al. 1992). Another, less restrictive, method is AOR, which applies to job shops (Li et al. 1993)(Abumaizar and Svestka 1997). If disruptions are present, only operations directly, or indirectly, affected by a machine breakdown, are right-shifted during execution. This method was in (Subramaniam and Raheja 2003) extended to cover more types of disruptions. In (Bean et al. 1991) and (Akturk and Gorgulu 1999), the authors propose rescheduling methods to match-up with the offline schedule at a certain time in the future, if unforeseen disruptions occur. A partial schedule is generated in (Wu et al. 1999), where some decisions are left to be decided on during execution. In (Abumaizar and Svestka 1997) and (Vieira et al. 2000), total rescheduling is used, where all remaining operations are rescheduled in response to a disturbance. In Paper 2, a predictive-reactive method is presented, whereas a predictive-reactive method is transformed to a purely predictive approach in Paper 3.

3.1.3 Reactive Scheduling

Reactive scheduling is solely performed online without a schedule. Mainly, dispatching rules are used to decide the execution sequence of operations on resources (Tay and Ho 2008). In (Jayamohan and Rajendran 2000), good performance is achieved in terms of minimizing the maximum tardiness when two rules are combined, namely the earliest due date and the longest remaining processing time first. Several priority dispatching rules are evaluated for a job shop, with due date objectives, to study the best performing approaches in (Chiang and Fu 2007).

3.1.4 Schedule Quality

The quality of a baseline schedule is evaluated by comparing it with the realized schedule, which can considerably deviate from the baseline schedule due to disturbances. One type of quality measure that regards the difference in objective value, such as makespan or mean tardiness between schedules, is referred to as a robustness measure. A schedule is said to be *robust* if the objective value does not deteriorate as a result of disturbances in the system. Another type of quality measure concerning stability compares the sequences in the baseline and realized schedule by e.g. measuring deviations in start time, sequence deviation etc. A schedule is said to be *stable* if the realized schedule does not deviate from the baseline schedule in the presence of disruptions. In the following chapter, the measures of robustness and stability concerned in this thesis are introduced. But first in this chapter, AOR and RSR are studied more extensively.

3.2 Affected Operations Rescheduling

In (Abumaizar and Svestka 1997), the authors propose a predictive-reactive rescheduling strategy for the job shop environment, considering machine breakdowns as disruptions. Initially, a deterministic, minimum time schedule is obtained, resembling the predictive part. The reactive part considers an algorithm based on a binary tree, where the operation affected by the machine breakdown is chosen as the root node. For job shops, each node will have at most two emanating branches, as described for the disjunctive graph in Section 2.3. The left branch of a node in the binary tree, is considered as the job branch, whereas the right branch as the machine branch. Hence, for any node, the succeeding node in the left branch, is the succeeding operation in the job sequence. The node in the right branch is the succeeding operation in the machine sequence, specified in the initial schedule. The basic concept is to go through the binary tree, and delay the start of affected operations once disruptions occur. The operation delay equals the time needed for the precedence constraints to be fulfilled, as well as to preserve the machine sequence. Thus, compared to RSR, only the start of operations directly affected or indirectly affected by a machine breakdown are delayed. Hence, the approach is called Affected Operations Rescheduling.

However, identical results can be generated completely offline, by approaching the problem as a deterministic job shop problem, and applying the disjunctive graph.

The approach presented in Paper 3, is based on results in the scheduling literature, where scheduling using disjunctive graphs has been extensively studied (Jain and Meeran 1999). The disjunctive graph was introduced in (Roy and Sussmann 1964), and more recently used for scheduling in e.g. (Kan 2012; Blażewicz et al. 2000). A small example, with two robots working closely in parallel, is used to illustrate the procedure. A common workspace, where collisions potentially can occur, is modeled as a unit capacity resource. The job sequence for each robot is modeled in the disjunctive graph depicted in Fig. 3.2. One robot has three operations to perform, while the other should perform two operations. Each node, except for the source and sink node, corresponds to an operation. As explained in Section 2.3, the processing time for an operation is modeled as weights on the emanating arcs from the corresponding node. For example, operation O_{11} has processing time 5. Hence the outgoing arc from node 11 has weight 5.

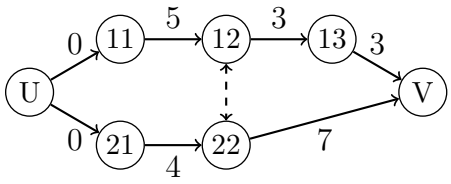


Figure 3.2: Disjunctive graph $G(\mathcal{N}, \mathcal{A}, \mathcal{E})$ with conjunctive arcs (solid) and disjunctive arc (dashed). The disjunctive arc corresponds to a common workspace.

The minimum time schedule is obtained by choosing the direction of the disjunctive arc such that the shortest longest path from the source node U to sink node V is achieved. This corresponds to node 12 executing prior to node 22, which results in a makespan equal to 15. In this quite trivial example with only one resource, a delay in 12 will affect both 13, which is the succeeding operation in the job sequence, as well as 22 which is the succeeding operation in the resource sequence. This approach is applicable to any job shop. The resulting graph, after disjunctive arcs are resolved and transitive relations removed, contains the partial execution order between nodes connected with arcs. For example, $\mathcal{P}_j$, denoting the index set of preceding operations to O_j, is in this case equal to nodes with emanating arcs pointing to node j. Hence, the preconditions in (2.1), ensure that only affected operations are delayed if disruptions are present.

RSR is often used as a benchmark when evaluating the performance of rescheduling methods proposed in the literature (Subramaniam et al. 2005; He and Sun 2013; O'Donovan et al. 1999). Next, RSR will be introduced by extending the example in Fig. 3.2. The disjunctive graph will be used to point out the difference between AOR and RSR.

3.3 Right-Shift Rescheduling

In RSR, all operations are globally right-shifted when disruptions occur. This implies that an operation has to wait for all preceeding operations in the schedule to be completed, before the operation can start to execute. The approach in Section 2.2.3,

where event-based conditions are obtained, results in the same system behavior as RSR. That is, before the relaxation is performed.

The example depicted in Fig. 3.2 will be used to show the difference between AOR and RSR. In order to prohibit operations to start before all prior operations are completed, early and late start times for operations are introduced. The early start time for a node i, corresponds to the longest path from the source node to that specific node, i.e. $\ell_{U,i}$. The late start time for a node i, is defined as the makespan subtracted with the longest path from that specific node to the sink node, i.e. $T - \ell_{i,V}$. In Fig. 3.3, early and late start times for each node is depicted in the boxes next to each node. To restrict an operation to start before prior operations are completed, an arc has to added between two nodes if the late start time plus the processing time is less than, or equal to, the early start of another node. The result is depicted in Fig. 3.3. Note that transitive relations have been removed. Initially, arcs between (11,22) and (21,13) were also added. Since the graph contains the path $11 \rightarrow 12 \rightarrow 22$, the arc between 11 and 22 is redundant.

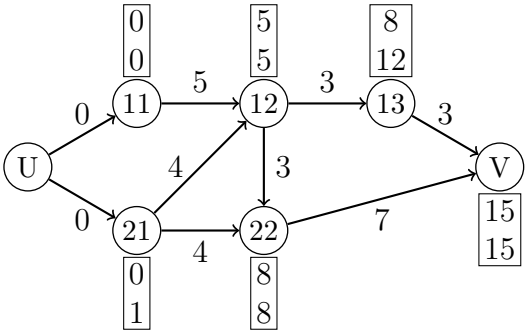


Figure 3.3: The resulting graph for the RSR approach.

In order to analyze the effect of adding arcs to the disjunctive graph, to fulfill the requirements of RSR, the time-optimal schedule is depicted in Fig. 3.4. Based on Fig. 3.2, in AOR, the only predecessor to 12 is 11 (since the resolved arc between 12 and 22 emanates from 12). This corresponds to operations O_{11} and O_{12} in the schedule, which belong to the same job. However, for RSR, 21 is added as a predecessor to 12 in Fig. 3.3. Hence, O_{12} cannot start before both O_{11} and O_{21} are completed. If an operation is delayed, this approach delays the immediate successors and the delay is propagated through the graph. However, if slack exists, the delay is diminished as it propagates through the schedule.

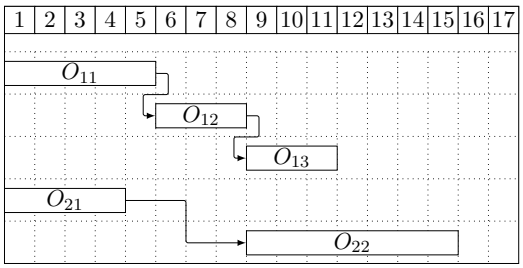


Figure 3.4: The time-optimal schedule.

The total slack, defined in (2.6) in Section 2.3, expresses the amount of time that an operation can be delayed without affecting the makespan. The total slack can also be expressed as the late start time, subtracted with the early start time. When arcs are added between nodes, the late start might be affected. Before the addition of arc $(21, 12)$ in Fig. 3.3, the late start was equal to 4. Thus, the total slack was, in this case, also equal to 4 since the early start time equals 0. Due to idle time between O_{21} and O_{22} in Fig. 3.4, O_{21} can be delayed with 4 time units without affecting the makespan. However, in RSR, the late start time and total slack for O_{21} equal 1. The total slack decreases as a result of adding successors, if the late start of the new successor is smaller than the late start of the other successors. This is due to an increase in the longest path which, according to (2.6), results in a decrease of the total slack. For O_{21}, a delay greater than 1 delays the start of O_{12}, which affects the makespan. In Paper 3, a proof is posed to show that AOR always performs better than, or equal to, RSR in the face of disruptions.

The illustrative example presented shows that idle time can be used to protect the schedule against disruptions. Idle time in deterministic, minimum time schedules is due to operations sharing resources and, as a consequence, operations have to wait for a common resource to become available. In redundancy-based methods, where addition slack is inserted to the schedule, a non-optimal makespan is allowed, in pursuance of a robust and/or stable schedule.

3.4 Summary

In this chapter, the rescheduling framework was introduced. Different strategies to implement protection against disruptions were presented as well as examples of underlying methods. Two predictive-reactive methods were studied more in-depth, namely Affected Operations Rescheduling and Right-Shift Rescheduling. In this thesis, Paper 2 concerns a predictive-reactive approach, while a predictive-reactive approach is formulated as a completely predictive method in Paper 3. Furthermore, proactive scheduling is considered in Paper 4. The quality of the resulting schedules in terms robustness and stability is briefly mentioned.

Next, robustness and stability measures are defined. Also, existing surrogate measures of robustness and stability are presented together with a proposed surrogate stability measure.

Chapter 4

Robustness and Stability

For deterministic systems, the realized schedule is equal to the initial schedule, developed offline. However, if disruptions are present and the system thus stochastic, the realized schedule will most certainly deviate from the initial schedule. When protection against disruptions is embedded in the schedule, the aim is for the realized schedule to adequately imitate the baseline schedule. To assess the quality of a baseline schedule, different measures of robustness and stability have been proposed. In this chapter, two frequently used measures for robustness and stability are introduced. Then, surrogate measures used in the literature on rescheduling are presented, after which a new surrogate measure is proposed. A benchmark problem is used to evaluate the quality of the different measures when disruptions are present.

4.1 Quality Measures

Measures of robustness and stability are commonly used to evaluate the quality of schedules in rescheduling (Goren and Sabuncuoglu 2008). These measures were introduced in Section 3.1.4. In this thesis, robustness is measured in terms of makespan delay between the baseline and the realized schedule. The average start time deviation between the baseline and the realized schedule is used to measure stability when the redundancy-based approach is addressed, whereas sequence deviations are otherwise regarded. Next, quality measures for makespan delay and average start time deviation are formally defined.

4.1.1 Robustness measure

A common robustness measure in the literature on rescheduling is given by the expected delay in makespan (Jorge Leon et al. 1994). Let $M_0(S)$ denote the makespan of the baseline schedule S. The actual makespan in the realized schedule is denoted by $M(S)$, which is a random variable. Hence, the delay in makespan can be expressed as $M(S) - M_0(S)$. Since $M_0(S)$ is deterministic, the expected makespan delay is given by

$$E[M(S)] - M_0(S) \tag{4.1}$$

As the difference in (4.1) decreases, the robustness increases.

4.1.2 Stability measure

For a schedule to be stable, the realization of the schedule should not deviate from the baseline schedule in the presence of disruptions. The weighted expected absolute deviation in start times, between the realized schedule and the baseline schedule, is selected as the stability measure (Lambrechts et al. 2011). As defined in Section 2.3, $\mathcal{N}$ denotes the index set of operations, $|\mathcal{N}| = n$, whereas w_i is the weight of operation O_i. The random variable T_i represents the start time of O_i and $E[T_i]$ is the expected value of the start time T_i in the realized schedule. The planned start time in the baseline schedule is denoted as t_i. The stability measure is given by

$$\sum_{i \in \mathcal{N}} w_i |E[T_i] - t_i| \tag{4.2}$$

Unit weights $w_i, i \in \mathcal{N}$, are used in this thesis. As the deviation in start times in (4.2) decreases, the stability increases. Due to tractability problems where the effect of a disruption depends on the outcome of all previous disruptions, surrogate measures are used to evaluate the quality. Next, existing surrogate measures are presented, as well as a suggested surrogate measure.

4.2 Surrogate Measures

In the rescheduling literature, several surrogate measures of robustness and stability have been proposed (Sabuncuoglu and Goren 2009)(Jensen 2003)(Al-Hinai and ElMekkawy 2011). The measures can either be based on the assumption that no information regarding the disruptions is known, or that some knowledge exists such as probability density functions describing the uncertainties. Methods using surrogate measures for stability and/or robustness, either embed the measure into the decision process or in a post-process step to evaluate the performance of the schedule. In this section, existing slack-based surrogate measures are presented as well as a proposed surrogate measure.

4.2.1 Existing measures

In (Jorge Leon et al. 1994), a surrogate robustness measure, based on the average total slack, is presented. Evaluations show a high correlation between robustness and average total slack, such that an increase in average total slack results in a system less sensitive to disruptions. The proposed measure is given by

$$RM_1 = \frac{1}{n} \sum_{i \in \mathcal{N}} f_i \tag{4.3}$$

where f_i is the total slack given by (2.6). In (Hazır et al. 2010), a surrogate robustness measure is presented, based on a function that has diminishing returns per extra unit slack. With this measure, it is more beneficial to add slack to an operation with

little slack, compared to an operation with large slack. The robustness measure is expressed as

$$RM_2 = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{f_i} e^{-j},\ f_i \in \mathbb{Z} \tag{4.4}$$

In (Hazır et al. 2010), the number of successors to each operation is included in the first sum, which makes it more favorable to add slack to an operation with many successors. Since the goal is to compare different types of measures, where the result of introducing the exponential term in the measure is of interest, unit weights are considered, in correspondence with the other measures. The measure in (4.3) is applicable to both discrete and continuous time formulations. In Section 4.3.1, the presented measures are included in a problem formulation, used to evaluate their quality and computational efficiency. Due to the inclusion of energy in Section 5.2.1, a continuous-time formulation is proposed. Hence, the robustness measure in (4.4) is reformulated. When the second sum is evaluated using an integral, the continuous-time measure is given by

$$RM_3 = \frac{1}{n}\sum_{i=1}^{n}\int_0^{f_i} e^{-t}dt = \frac{1}{n}\sum_{i=1}^{n}\left(1 - e^{-f_i}\right),\ f_i \in \mathbb{R} \tag{4.5}$$

The measures in (4.3) and (4.5) incorporate the total slack for operations. They are thus considered as robustness measures, since the total slack is related to robustness. However, the measures can easily be transformed to stability measures, by exchanging the total slack, f_i, with the free slack, s_i, given by (2.5). This property is general. When exchanging total slack with free slack, the measure will generate a stability criterion that evaluates start time deviations. Let $SM_i, i \in \{1,2,3\}$ denote a stability measure corresponding to $RM_i, i \in \{1,2,3\}$, but where the total slack has been replaced with free slack.

For delays larger than the free slack, the start of succeeding operations are affected. Next, a stability measure is proposed that will impose a cost for delays of that extent.

4.2.2 Proposed stability surrogate measure

Let the delay of an operation O_i be represented by a random variable with known probability distribution. The duration of O_i is denoted as $d_i = d_i^0 + \delta_i$, $d_i \in \mathbb{R}$, where d_i^0 is the nominal duration and δ_i is a realization of the delay. If $\delta_i > s_i$, the delayed operation will delay the start of a succeeding operation. A stability measure, which imposes a cost if the delay of an operation exceeds its free slack, is proposed. Let $p(\delta_i)$ denote the probability density function, describing the likelihood that operation O_i has a delay δ_i. The suggested cost is given by

$$c(s_i) = \int_0^{\infty} p(\delta_i) \cdot \max(0, \delta_i - s_i)\ d\delta_i \tag{4.6}$$

Delays with uniform distribution is considered, $\mathcal{U}(\delta_i^{min}, \delta_i^{max})$. In this case, $p(\delta_i) = \frac{1}{\delta_i^{max} - \delta_i^{min}}$ for $\delta_i \in (\delta_i^{min}, \delta_i^{max})$, and 0 otherwise. Hence, for a uniform probability

density function, the cost in (4.6) is given by

$$c(s_i) = \int_{\delta_i^{min}}^{\delta_i^{max}} \frac{1}{\delta_i^{max} - \delta_i^{min}} \cdot \max(0, \delta_i - s_i)\ d\delta_i \tag{4.7}$$

A minimum delay equal to 0 is considered, i.e. $\delta_i^{min} = 0$. The integral in (4.7) equals 0 for $\delta_i \leq s_i$. Hence, for delays $\delta_i > s_i$, the cost can be expressed as

$$c(s_i) = \frac{1}{\delta_i^{max}} \int_{s_i}^{\delta_i^{max}} \delta_i - s_i\ d\delta_i = \frac{1}{2\delta_i^{max}} \Big(\delta_i^{max} - s_i\Big)^2$$

To conclude, the cost for an operation O_i is given by

$$c(s_i) = \begin{cases} \frac{1}{2\delta_i^{max}} \Big(\delta_i^{max} - s_i\Big)^2, & \text{if } \delta_i^{max} > s_i \\ 0, & \text{otherwise} \end{cases} \tag{4.8}$$

The proposed stability measure is given by

$$SM_4 = \frac{1}{n} \sum_{i \in \mathcal{N}} c(s_i) \tag{4.9}$$

The stability measure SM_4 is reformulated to a robustness measure, RM_4, by replacing the free slack with total slack. For a uniform probability distribution, the resulting measure in (4.9) is quadratic. In the literature on rescheduling, uniform distributions are commonly used for processing time delays. Other distributions considered are e.g. the beta distribution or the log-normal distribution. The use of these distributions results in much more complex expressions. A quadratic measure is preferable considering computation time during optimization.

The measures RM_1/SM_1 and RM_3/SM_3 are linear and exponential respectively while the proposed measure is quadratic. Thus, the computation time for generating schedules should also be taken into consideration while comparing the different measures. Next, the measures are evaluated.

4.3 Evaluation of Surrogate Measures

In this section, the surrogate measures are evaluated based on the quality of the schedules obtained with the different surrogate measures included as objectives. For the robustness measures, the makespan delay for random processing time delays, as well as the computation time, are considered. Similarly, for the stability measure, the average start time deviation and computation time are studied when disruptions are present. First, an optimization model is formulated to perform the analysis.

4.3.1 Optimization model

Based on the disjunctive graph presented in Section 2.3, a mixed integer nonlinear programming (MINLP) model is formulated for the job shop problem (Wigström and Lennartson 2013). The MINLP formulation is derived based on the precedence

relations originating from job sequences, as well as mutual exclusion constraints, for the pairwise execution order for operations requiring the same resource. Due to the total slack, included in the robustness measures, constraints are formulated for the longest path. Also, free slack is included into the formulation due to the stability measures. Each operation $O_i, i \in \mathcal{N}$, has a start time t_i, duration d_i, free slack s_i and longest path ℓ_i. For all $i \in \mathcal{N}$, $t_i,\ d_i,\ s_i,\ \ell_i \in \mathbb{R}$. The preconditions for operations in job sequences affect the timing as

$$t_i + d_i + s_i \leq t_j \quad \forall\,(i,j) \in \mathcal{A} \tag{4.10}$$

and the longest path of an operation is always longer than its duration together with the succeeding operation's longest path

$$\ell_j + d_i \leq \ell_i \quad \forall\,(i,j) \in \mathcal{A} \tag{4.11}$$

As for the mutual exclusion constraints, for each pair $(i,j) \in \mathcal{E}$, introduce a boolean variable b_{ij}, where $b_{ij} = 1$ if operation O_i is before O_j, and $b_{ij} = 0$ otherwise. The timing is modeled as

$$\begin{aligned} t_i + d_i + s_i &\leq t_j + M(b_{ij} - 1) \\ t_j + d_j + s_j &\leq t_i + M b_{ij} \qquad \forall\,(i,j) \in \mathcal{E} \end{aligned} \tag{4.12}$$

where M is a sufficiently large constant. The longest path constraint becomes

$$\begin{aligned} \ell_j + d_i &\leq \ell_i + M(b_{ij} - 1) \\ \ell_i + d_j &\leq \ell_j + M b_{ij} \qquad \forall\,(i,j) \in \mathcal{E} \end{aligned} \tag{4.13}$$

The optimization model has multiple objectives corresponding to either makespan $\boldsymbol{T}$ and robustness $\boldsymbol{R}$, or makespan $\boldsymbol{T}$ and stability $\boldsymbol{S}$ depending on whether a robust or stable schedule is developed.

$$\min(\boldsymbol{T}, \boldsymbol{R}) \ \text{ or } \ \min(\boldsymbol{T}, \boldsymbol{S}) \tag{4.14}$$

The makespan $\boldsymbol{T}$ is given by $\max(t_i + d_i + s_i), \forall i \in \mathcal{N}$. The robustness $\boldsymbol{R}$ is defined by one of the robustness measures presented in Section 4.2, i.e.

$$\boldsymbol{R} \triangleq -RM_1, -RM_3 \text{ or } RM_4 \tag{4.15}$$

Note that RM_1 and RM_3 have negative signs since these should be maximized. The stability $\boldsymbol{S}$ is defined by one of the corresponding stability measures.

$$\boldsymbol{S} \triangleq -SM_1, -SM_3 \text{ or } SM_4 \tag{4.16}$$

The performance of the surrogate measures are evaluated based on the presented optimization model. A measure might be more suitable as a robustness measure, compared to a stability measure and vice versa. Hence, this possibility is also taken into consideration during the analysis. Before the evaluations begin, the benchmark problem is presented.

4.3.2 Benchmark problem

The well-known Fisher Thompson benchmark problem is used to compare the measures presented in Section 4.2 (Taillard 1993). The problem instance considered is the 6×6 problem, where each job has 6 operations, each processed on 6 different resources, giving a total of 36 operations.

During the following analysis, the minimum makespan of a schedule is denoted as T^*. Let R_{T^*} and S_{T^*} denote the reference measures for the robustness and stability of the minimal makespan schedule. In the same way, R^* denotes minimal robustness, while T_{R^*} and S_{R^*} denote the makespan and stability of the minimal robustness schedule etc. The effect on makespan for random duration delays is studied for different upper bounds on makespan, equal to $(x + 100)\%$ T^*, $x \in \{10, 12.5, \ldots, 20\}$. Delays δ_i are uniformly distributed, $\delta_i \in (0, \delta_i^{max})$, where $\delta_i^{max} \in \{0.2d_i^0, 0.3d_i^0, 0.4d_i^0, 0.5d_i^0\}$.

All optimization was run on a Windows 7 64 bit system with 2.67 [GHz] Intel Core i5 CPU and 8 [GB] RAM. The optimization problem was modeled in AMPL. The solvers used were CPLEX, IPOPT and BONMIN.

4.3.3 Comparison of robustness measures

As a first step to compare the robustness measures, a minimum time schedule is initially developed where the makespan is given by T^*. Then, robust schedules are generated by minimizing $\boldsymbol{R}$ and introducing an upper bound for the makespan $\boldsymbol{T}$. For each choice of makespan upper bound, as well as δ_i^{max}, delays are randomly generated 20 times. A total of 400 instances are studied, 100 for each δ_i^{max}. The makespan delay, i.e. the difference between the makespan of the disrupted schedules, compared to the makespan T_{R^*} of the baseline schedules, is retrieved. This procedure is performed for all robustness measures in (4.15). In Section 4.1.1, the makespan of the realized schedule S is denoted as $M(S)$. Hence, the makespan of the disrupted schedules resembles outcomes of $M(S)$. Note that $T_{R^*} = M_0(S)$.

In Fig. 4.1, a bar plot representing the average delay in makespan for different values on δ_i^{max} is depicted. For each value, three bars are depicted, representing the different robustness measures used to generate robust schedules. The average delay in makespan is quite similar for all robustness measures. For greater duration delays, RM_4 results in an average makespan delay that is slightly smaller compared to the results for RM_1 and RM_3.

In Fig. 4.2, the computation time is depicted as a function of the upper bound on makespan, ranging from an increase in 2.5% to 40%. The CPU time for minimizing robustness measure RM_3 is much worse than for the other measures. This is anticipated due to the exponential terms in RM_3, whereas the terms in RM_1 are linear and quadratic for RM_4. For smaller upper bounds on makespan, the computation time for RM_1 is shorter than for RM_4. However, the opposite holds for greater upper bounds on makespan. At a 25% makespan increase, the computation time for RM_3 drops. A possible reason to this behavior is that, when the increase in makespan is large, the exponential terms in the measure will become so small that solutions

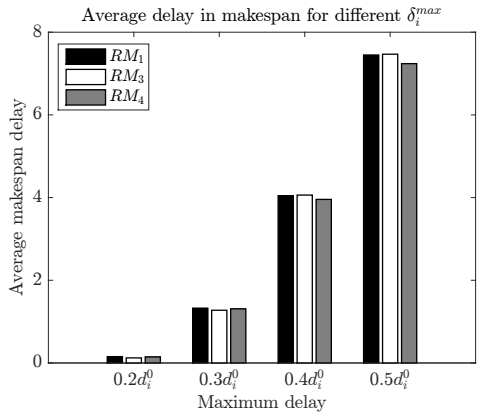


Figure 4.1: The average delay in makespan for schedules generated by RM_1, RM_3 and RM_4 with a makespan upper bound equal to $(x + 100)\%$ T^*, $x \in \{10, 12.5, \ldots, 20\}$. The makespan in the delayed schedule is compared with the makespan in the baseline schedule. Delays δ_i are generated from the interval $(0, \delta_i^{max})$, where $\delta_i^{max} \in \{0.2d_i^0, 0.3d_i^0, 0.4d_i^0, 0.5d_i^0\}, d_i^0$: deterministic duration, $i \in \mathcal{N}$.

close to the relaxed root node are found. Thus, the branch and bound algorithm will terminate quite fast.

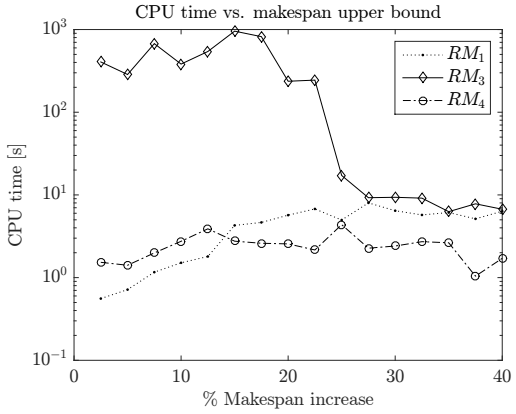


Figure 4.2: The computation time (CPU time) for optimizing the robustness measures as a function of the increase in makespan equal to $(x + 100)\%$ T^*, $x \in \{2.5, 5, \ldots, 40\}$.

Based on the comparison of robustness measures, a conclusion can be drawn that both RM_1 and RM_4 results in quite similar performance, both in makespan delays and computation time. The makespan delay for RM_3 is comparable to that of the other measures. However, the computation time is much worse when generating robust schedules with an upper bound on makespan less than $1.25T^*$.

4.3.4 Comparison of stability measures

Mainly the same procedure is repeated for the comparison of stability measures. A time-optimal schedule is first generated to determine the minimum makespan

T^*. As a next step, stable schedules are developed, by individually minimizing the stability measures in (4.16), together with an upper bound on makespan. For each choice of makespan upper bound, as well as δ_i^{max}, delays are randomly generated 20 times. A total of 400 instances are studied, 100 for each δ_i^{max}. The start time deviations between operations in the disrupted schedules and the baseline schedules are retrieved. Referring back to Section 4.1.2, the outcome of T_i corresponds to the start time of operation O_i in the realized, i.e. disrupted, schedule. The planned start time, t_i, corresponds to the start time in the stable baseline schedule.

In Fig. 4.3, the average deviation in start times for the different stability measures, is depicted. The execution order of schedules generated with SM_3 is fixed, due to computational complexity. When a MINLP problem was attempted to be solved to maximize SM_3, the optimization procedure was terminated due to long execution times. Instead, a nonlinear programming (NLP) problem was solved, enforcing an execution order equal to the minimum makespan schedule. Based on the results, these sequences give fairly good performance in terms of start time deviation. The average deviation in start time for SM_1 is outperformed by the other measures. Since the average free slack is considered for SM_1, a solution can place all free slack after one single operation, whereas the other operations are left without protection. As a consequence of not distributing the free slack, the resulting schedule is more sensitive to delays.

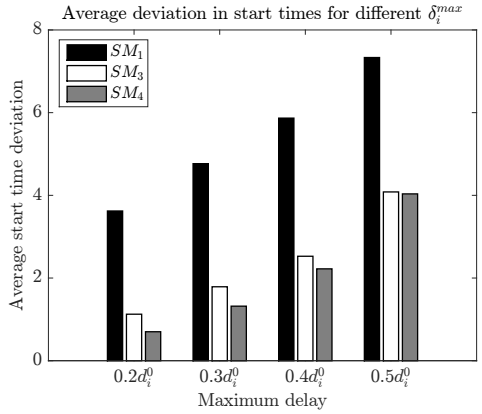


Figure 4.3: The average deviation in start times for schedules with delays compared to baseline stable schedules based on stability measures SM_1, SM_3 and SM_4. A makespan upper bound equal to $(x + 100)\%\ T^*,\ x \in \{10, 12.5, \ldots, 20\}$ is used. Delays are in the range $(0, \delta_i^{max})$ where $\delta_i^{max} \in \{0.2d_i^0, 0.3d_i^0, 0.4d_i^0, 0.5d_i^0\}$, d_i^0: deterministic duration, $i \in \mathcal{N}$.

The computation time for different upper bounds on makespan, ranging from an increase in 2.5% to 40%, is depicted in Fig. 4.4. The computation time for SM_3 is not included since the optimization was terminated, due to long computation times, $> 10^3$ s, for makespan increases $> 5\,\%$ for this measure. The performance of SM_4 exceeds SM_1 in terms of average start time deviation. However, the computation time for SM_4 is worse, but overall still quite short.

The proposed measure performs better as a stability measure. This is due to the local property of free slack, which is connected to one operation. If the delay is

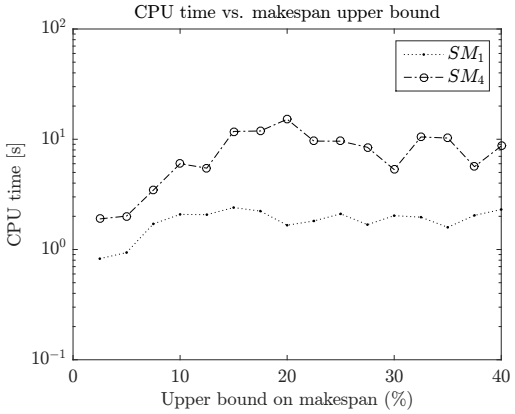


Figure 4.4: The computation time (CPU time) for optimizing the stability measures as a function of the increase in makespan equal to $(x + 100)\%\ T^*,\ x \in \{2.5, 5, \ldots, 40\}$.

smaller than the free slack, succeeding operations will not be affected by the delay. However, for total slack, an operation with a delay smaller than the total slack, will still reduce the total slack for succeeding operations that are directly or indirectly connected to the delayed operation.

4.4 Summary

Measures of robustness and stability were introduced in this chapter. Existing surrogate measures for robustness and stability in the literature on rescheduling were presented together with a proposed surrogate measure. An optimization model was stated for which robust and stable schedules can be obtained. The surrogate measures were evaluated by using a benchmark problem. The makespan delay was used to measure robustness, while the start time deviation was considered in the stability measure. According to the results, the proposed stability measure performs better as a stability measure compared to a robustness measure.

Next, an objective, considered when developing energy efficient schedules, is introduced. The optimization model presented in this chapter is extended to also include energy consumption. Then, a conflict between energy, stability and robustness is high-lighted.

Chapter 5

Energy Efficient Scheduling

In order to generate energy-efficient schedules, much effort has been devoted to reducing the energy consumption of industrial robots, due to their extensive use in automated manufacturing systems. In (Vergnano et al. 2012), a method to reduce the energy consumption for robots is proposed by reducing, and often eliminating, idle time between operations. The energy consumption for robots is embedded into the scheduling model. Each operation is equipped with an energy consumption signature parameterized by its execution time. In (Wigström et al. 2013), an extension was introduced, resulting in further improvements on energy consumption by using dynamic time scaling of the robot trajectories. The reduction of peak energy consumption in a flexible flow shop was studied in (Bruzzone et al. 2012). In (Riazi et al. 2017), results show that energy can be saved, even without changing the operation sequences. An assumption of a convex energy consumption signature is made. Shared zones for robots give rise to gaps, i.e. idle time, in the schedule. By extending the execution time for a waiting robot, a reduction in energy usage is achieved. Hence, idle time in the resulting schedule is reduced.

Clearly, a conflict exists between redundancy-based techniques used in rescheduling and energy optimization, since the former depends on idle time while the latter benefits from reducing, and often, eliminating it. In this chapter, the conflicts that arise when these techniques are combined, are highlighted.

5.1 Energy Consumption Measure

In this thesis, the energy consumption signature for an operation, parameterized by its duration, is based on an experimentally derived energy consumption signature for an industrial robot. During experiments, a specific motion is repeated using different execution times, while measuring the energy consumption. The resulting measures of energy consumption are depicted in Fig. 5.1.

Let d_i^0 represent the nominal, unextended duration of operation O_i, while $e_i(d_i)$ denotes its energy consumption for a duration d_i. The recorded energy function for the robot is generalized in the following way: (i) at nominal execution time d_i^0, operations consume e_i^0 units of energy; (ii) at twice the original execution time, $2d_i^0$, energy is reduced by 20%; (iii) at $3.5d_i^0$, the energy consumption increases in a purely

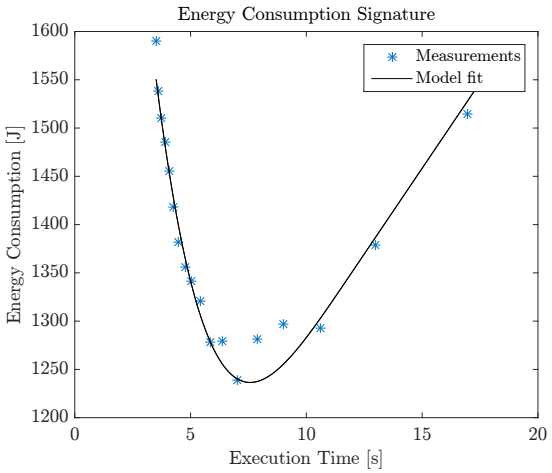


Figure 5.1: The energy consumption signature for a robot given by the energy consumption as a function of execution time. The stars correspond to measurements of energy consumption for different execution times. The solid line is the model fit of the measurements.

linear fashion due to friction; (iv) at $5d_i^0$, the energy consumption is once again e_i^0. The following convex model is posed for the energy functions

$$e_i(d_i) = \begin{cases} \alpha(3.5d_i^0 - d_i)^4 + \beta d_i + \gamma, & \text{if } d_i \leq 3.5d_i^0 \\ \beta d_i + \gamma, & \text{otherwise} \end{cases} \tag{5.1}$$

The above specification gives roughly $\alpha = e_i^0/(3.32d_i^0)^4$, $\beta = e_i^0/(12.4d_i^0)$ and $\gamma = e_i^0/1.675$. The resulting fit for the model is also shown in Figure 5.1. The root mean square error of the model fit is 19.66 J and the worst case error is 43.68 J.

The derived signature is adopted for all operations. In reality, each operation has a unique energy consumption signature. However, any convex function describing the energy consumption, as a function of the execution time of an operation, could be used to demonstrate the concept of this method. The results can be extended to other resource types if their energy signature can be approximated with a convex function, such that a unique minimum exists. An assumption is made, where the nominal execution time is considered to be shorter than that achieving the minimum energy consumption. In this case, an increase in execution time implies a decrease in energy consumption. Otherwise, eliminating slack is not energy efficient.

The measure of energy consumption is considered to be the sum of the convex energy functions, i.e.

$$\sum_{i \in \mathcal{N}} e_i(d_i)$$

At this point, measures for robustness, stability and energy have been presented. Hence, the potential conflict can be evaluated. In the following section, the previously posed optimization model is extended to also include energy. As a result, trade-off analysis can be performed.

5.2 Conflict Between Energy, Stability and Robustness

When the energy measure, derived in Section 5.1, is used together with the constraints (4.10)-(4.13) in Section 4.3.1, an energy-efficient schedule is developed. As previously mentioned, slack is reduced on behalf of longer execution times in this case. Hence, a conflict emerges when energy minimization is combined with slack-based rescheduling techniques. An optimization formulation is posed to analyze the trade-off.

5.2.1 Optimization model

Let the makespan, stability and robustness be defined as in Section 4.3.1. The energy consumption of the system is defined by the energy measure, i.e. a sum of convex energy functions

$$\boldsymbol{E} \triangleq \sum_{i \in \mathcal{N}} e_i(d_i) \tag{5.2}$$

where $e_i(d_i)$ is the energy consumption signature for operation O_i given by (5.1). The objectives in (4.14) are thus extended to also include the energy consumption measure in (5.2). Together with constraints (4.10)-(4.13), the compact representation of the optimization model is given by

$$\begin{aligned}
&\min \quad (\boldsymbol{T}, \boldsymbol{R}, \boldsymbol{S}, \boldsymbol{E}) \\
&s.t. \\
&\quad t_i + d_i + s_i \leq t_j && \forall\,(i,j) \in \mathcal{A} \\
&\quad \ell_j + d_i \leq \ell_i && \forall\,(i,j) \in \mathcal{A} \\
&\quad t_i + d_i + s_i \leq t_j + M(b_{ij} - 1) && \forall\,(i,j) \in \mathcal{E} \\
&\quad t_j + d_j + s_j \leq t_i + M b_{ij} && \forall\,(i,j) \in \mathcal{E} \\
&\quad \ell_j + d_i \leq \ell_i + M(b_{ij} - 1) && \forall\,(i,j) \in \mathcal{E} \\
&\quad \ell_i + d_j \leq \ell_j + M b_{ij} && \forall\,(i,j) \in \mathcal{E} \\
&\quad t_i, d_i, s_i, \ell_i \in \mathbb{R}^+ && i \in \mathcal{N} \\
&\quad b_{ij} \in \mathbb{Z} && i,j \in \mathcal{N}
\end{aligned} \tag{5.3}$$

where t_i, d_i, s_i, ℓ_i and b_{ij} are decision variables.

5.2.2 Trade-off analysis

Based on the comparisons of measures performed in Section 4.3, a robustness measure and stability measure are selected for the trade-off analysis. In Section 4.3.3, the average delay in makespan was quite similar for all robustness measures. RM_1 has a slightly shorter computation time for small makespan increases and, thus used for the trade-off experiments. Based on the results in Section 4.3.4, SM_4 is used as a stability measure, due to its exceeding performance. To conclude, $\boldsymbol{R} = -RM_1$ and $\boldsymbol{S} = SM_4$.

The optimization model is solved by using the ε-constraint method (Miettinen 2012). In this method, one of the objectives is selected to be optimized, while the rest of the objectives are formulated as additional constraints bounded from above. In this case, the robustness $\boldsymbol{R}$ is minimized, while introducing bounds on makespan, stability and energy. The constraints added to the model are given by

$$\begin{aligned} \boldsymbol{T} &\leq \varepsilon_T \\ \boldsymbol{S} &\leq \varepsilon_S \\ \boldsymbol{E} &\leq \varepsilon_E \end{aligned} \tag{5.4}$$

where $\varepsilon_T, \varepsilon_S$ and ε_E are constants. A Pareto front is generated by gridding the energy and stability and minimizing the robustness at each grid point, with $\varepsilon_T, \varepsilon_S$ and ε_E set to values given by the specific grid point. A 10% increase in makespan is allowed, i.e. $\varepsilon_T = 1.1T^*$, where T^* is the minimal makespan.

The energy grid is first determined, where the smallest value is obtained by minimizing the energy consumption subject to the makespan upper bound. The largest value equals the sum of the energy functions at nominal execution time, i.e. E_{T^*}. For each energy grid point, the stability grid is obtained. The smallest value is determined by minimizing the stability subject to the makespan upper bound, as well as an energy upper bound, equal to the current energy grid point. The largest value for the stability grid is obtained by minimizing the robustness, subject to the same upper bounds on makespan and energy. Thus, the robustness is minimized in each grid point of the 2-dimensional energy and stability grid with bounds on makespan, energy and stability, based on the values for that specific grid point.

The resulting Pareto front is depicted in Fig. 5.2. Each curve corresponds to a constant level of energy consumption. Based on the Pareto front, the following results are achieved

- Given a certain ε_E and ε_T, as $\boldsymbol{R}$ decreases $\boldsymbol{S}$ increases;
- Given a certain $\boldsymbol{R}$ and ε_T, as $\boldsymbol{S}$ decreases $\boldsymbol{E}$ increases;
- Given a certain ε_S and ε_T, as $\boldsymbol{R}$ decreases $\boldsymbol{E}$ increases.

As expected, low energy consumption corresponds to poor stability and robustness resulting in schedules more sensitive to delays in makespan and start time deviations. The Pareto front reaches smaller values for $\boldsymbol{R}$ and $\boldsymbol{S}$ as $\boldsymbol{E}$ increases. In other words, an increase in energy consumption enables more robust and stable solutions. The problem is symmetric, i.e. if either $\boldsymbol{S}$ or $\boldsymbol{E}$ is minimized with bounds on the other measures, the result would be the same.

With the convex energy model used, a unique minima of the energy consumed by an operation exists for a certain duration. A decrease in execution time corresponds to an increase in energy consumption if the energy minimum have already been reached and passed. In (2.5), the free slack increases as the duration decreases. Hence, a smaller value for SM_4 is achieved. For the total slack in (2.6), a decrease in duration corresponds to a shorter longest path. As a result, RM_1 becomes greater.

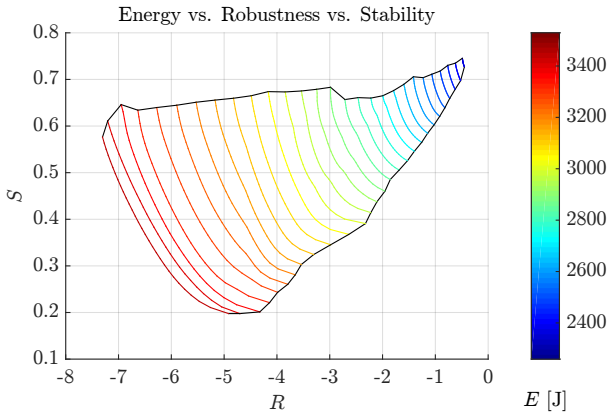


Figure 5.2: Pareto front for energy consumption ($\boldsymbol{E}$), robustness ($\boldsymbol{R}$) and stability ($\boldsymbol{S}$) with $\varepsilon_T = 1.1T^*$ and $\delta_i^{max} = 0.3d_i^0$. Each curve represents the trade-off between robustness and stability for a constant energy consumption [J].

Hence, smaller values for $\boldsymbol{R}$ and $\boldsymbol{S}$ are achieved. Also, a conflict between stability and robustness is evident. A small value for $\boldsymbol{S}$ comes at the cost of an increase in $\boldsymbol{R}$ and vice versa. For robustness, it is beneficial to add idle time at the end of the schedule. On the contrary, the stability gains from adding idle time in between operations, throughout the schedule.

Changing δ_i^{max} would only affect $\boldsymbol{S}$ which is defined by stability measure SM_4. An increase in δ_i^{max} would require extra free slack to keep the cost in (4.9) down if operation O_i is delayed. Hence, $\boldsymbol{S}$ increases as δ_i^{max} increases.

An increase in ε_T enables for solutions with lower energy consumption since further increments in execution time for operations are possible. That is, if the minimum of the energy consumption signatures have not yet been reached. A makespan increase that corresponds to the free slack s_i becoming greater than the maximum delay, δ_i^{max}, would result in a zero cost $c(s_i)$ for that operation. Hence, as ε_T increases, SM_4 would eventually reach 0. Note that this is based on the assumption on bounded delays due to delays with uniform distribution. The robustness measure, RM_1, does not consider delays and will hence increase as ε_T increases. The conflict between $\boldsymbol{E}$, $\boldsymbol{R}$ and $\boldsymbol{S}$ will terminate once the makespan is increased to a point where the energy consumption signatures for all operations have reached a minimum and the stability $\boldsymbol{S}$ has reached 0.

5.3 Summary

In this chapter, a measure for energy consumption was presented corresponding to a sum of energy signatures where each function is parameterized by the duration for the corresponding operation. This measure was then included in the optimization model previously presented. By minimizing the robustness and bounding the other objective values, a Pareto front was generated to show the trade-off between energy consumption, stability and robustness.

In the next chapter, a trajectory planning approach is presented, guaranteeing collision free trajectories for robots in a common workspace setting when the time at which a shared space becomes available can deviate from an initial schedule. In such scenarios, robust constraints can be used to ensure the possibility to stop before entering the common workspace.

Chapter 6

Robust and Energy Efficient Trajectory Planning

Due to the increase in automation and restricted space, robots can often be placed such that the workspace of one robot overlaps with the workspace of neighboring robots. This overlapping space, referred to as the common workspace, is a critical region where collisions might occur. Traditionally, collision-free trajectories are planned such that the time at which a robot enters the common workspace, is greater than the time at which a prior robot leaves the shared space. If a disruption occurs and the shared space is not yet available when the robot enters, a collision could potentially occur.

In this chapter, a method is proposed for which robust trajectories are obtained, guaranteeing under all circumstances collision-free paths. A clearance point is introduced along the path, where the occupancy of the shared space is evaluated. The velocity is restricted at this position, such that the robot is able to stop before entering the common workspace. A problem formulation is stated, where multiple objectives including final time and energy consumption are considered. The impact on the performance is analyzed, concerning the position and timing related to the clearance point.

6.1 Problem Formulation

Consider an n-degree-of-freedom robot with joint angles $\mathbf{q} \in \mathbb{R}^n$. Let $\mathbf{q}(s)$ denote a fixed path, given in joint space coordinates, where $s \in [0, 1]$ is a normalized scalar path coordinate and $\mathbf{q} : [0, 1] \to \mathbb{R}^n$. The path coordinate determines the spatial geometry of the path. Let the time t, path velocity $\dot{s}$ and path acceleration $\ddot{s}$ be functions of s, i.e. the problem is posed in space rather than in time. The path velocity is positive, i.e. $\dot{s}(s) \geq 0$ with boundary conditions

$$\dot{s}(0) = 0 \quad \dot{s}(1) = 0. \tag{6.1}$$

The derivatives of $\mathbf{q}(s)$ corresponding to joint velocities and accelerations are given by

$$\dot{\mathbf{q}}(s) = \mathbf{q}'(s)\dot{s}(s) \tag{6.2}$$
$$\ddot{\mathbf{q}}(s) = \mathbf{q}'(s)\ddot{s}(s) + \mathbf{q}''(s)\dot{s}^2(s) \tag{6.3}$$

where $(\,)' = d/ds$, such that $\mathbf{q}'$ and $\mathbf{q}''$ are the first and second spatial derivatives of the path respectively. The joint velocities and accelerations are constrained along the path

$$|\dot{\mathbf{q}}(s)| \leq \mathbf{v}^{\lim}(s) \tag{6.4}$$
$$|\ddot{\mathbf{q}}(s)| \leq \mathbf{a}^{\lim}(s), \tag{6.5}$$

where $\mathbf{v}^{\lim}$ and $\mathbf{a}^{\lim}$ are path varying bounds for joint velocities and accelerations respectively. The bounds have been made symmetric without loss of generality.

Constraints concerning collision avoidance are required due to a common workspace. The order in which robots use a shared space is assumed to be given. The perspective of the robot with lower priority is taken in this problem formulation. This robot cannot enter the shared space before the preceding robot exits. Let τ denote the time at which the common workspace becomes free to traverse. The boundary of the common workspace is located at $s = \beta$. A traditional collision avoidance constraint would state that $t(\beta) \geq \tau$.

Due to uncertainties in the system, the shared workspace might become available later than expected. In such a case, the robot might inadvertently traverse β before the common workspace is actually free, and hence, a collision could potentially occur. In this thesis, an approach guaranteeing collision-free trajectories is proposed. A clearance point α, $0 \leq \alpha \leq \beta$ is introduced. As the robot reaches the clearance point α, the availability of the common workspace is evaluated. The robot enters the common workspace if it is free. However, if the shared workspace is not yet available due to a disruption, the robot must be able to stop at β. The velocity at the clearance point is thus limited by

$$\dot{s}(\alpha) \leq v^{\max}(\alpha) \tag{6.6}$$

where $v^{\max}(\alpha)$ is the maximum velocity at α still ensuring that, if needed, the robot can stop before traversing β. Note, as the location of α approaches β, the more strictly bounded the velocity is due to the necessity to be able to stop at β. The earliest time at which the robot can traverse α is at time τ. Hence, we pose a timing constraint given by

$$t(\alpha) \geq \tau \tag{6.7}$$

The time $t(s)$ and path velocity $\dot{s}(s)$ are related as

$$t'(s) = \frac{dt(s)}{ds} = \frac{1}{\dot{s}(s)} \tag{6.8}$$

where $(\dot{}) = d/dt$, or on integral form as

$$t(s) = \int_0^s \frac{1}{\dot{s}(\sigma)} d\sigma \tag{6.9}$$

The boundary of $t(s)$, is given by

$$t(0) = 0 \quad t(1) = T \tag{6.10}$$

where T is the time at which the robot arrives at the end of its path. Multiple objectives corresponding to minimization of final time and energy consumption are considered,

$$\text{minimize} \quad \Big(T, E\Big) \tag{6.11}$$

Minimum final time is prioritized and energy consumption is thus secondary. To summarize, the compact representation of the mathematical model is given by

$$\begin{aligned}
\text{minimize} \quad & \Big(T, E\Big) \\
\text{subject to} \quad & t(0) = 0, \quad t(1) = T \\
& \dot{s}(0) = 0, \quad \dot{s}(1) = 0 \\
& t'(s) = 1/\dot{s}(s) \\
& \left.\begin{aligned} & |\mathbf{q}'(s)\dot{s}(s)| \leq \mathbf{v}^{\lim}(s) \\ & \Big|\mathbf{q}'(s)\ddot{s}(s) + \mathbf{q}''(s)\dot{s}^2(s)\Big| \leq \mathbf{a}^{\lim}(s) \end{aligned}\right\} \forall s \in [0,1] \\
& \dot{s}(\alpha) \leq v^{\max}(\alpha) \\
& t(\alpha) \geq \tau
\end{aligned} \tag{6.12}$$

where t, $\dot{s}$ and $\ddot{s}$ are optimization variables.

6.2 Minimum Time Analysis

Based on the problem formulation, both the position of α as well as the time τ impact the performance. In this section, an expression for the minimum time in which a path is traversed is derived, denoted as $T^*(\alpha, \tau)$. First, $\tau = 0$ is regarded, corresponding to the shared space being instantaneously available, i.e. $\tau = 0$. The final time expression is thereafter extended to hold for an arbitrary τ. Analysis is performed in the following section to study how α and τ affect the final time.

In our analysis, we rely heavily on the velocity profile of the minimum time solution and its properties. Let $\dot{s}^*(s)$ denote the velocity profile of the minimum time solution as a function of the path coordinate s. In (Bobrow et al. 1985), the results show that obtaining a minimum time trajectory includes selecting the acceleration profile that produces the largest possible velocity profile such that, at each point along the path, the velocity is not greater than the maximum velocity at which the actuators can hold the manipulator on the path. As a result, $\dot{s}^*(s)$ will always attain its maximal possible velocity for each point along the path, taking into account maximal accepted accelerations whilst satisfying the boundary conditions.

Due to the velocity bound at the clearance point in (6.6), $\dot{s}^*(s)$ is affected by the position of α. To distinguish between solutions for different α, $\dot{s}^*_\alpha(s)$ is introduced, denoting the minimum time velocity profile for an arbitrary $\alpha \in [0, \beta]$, see Fig. 6.1.

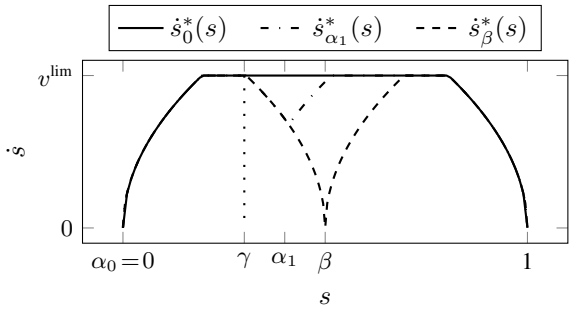


Figure 6.1: Three minimum time velocity profiles $\dot{s}^*_\alpha(s)$ for $\alpha \in \{0, \alpha_1, \beta\}$. The point $s = \gamma$ denotes the position where the robot starts to decelerate.

Note that, when $\alpha = \tau = 0$, the robot traverses the path without braking before traversing β, since constraints (6.6) and (6.7) are satisfied. This gives rise to an unrestricted solution, denoted as $\dot{s}^*_0(s)$ (since $\alpha = 0$), also illustrated in Fig. 6.1. If the clearance point is located at the boundary of the common workspace, i.e. $\alpha = \beta$, the robot must stop at β in case the shared space is assessed as occupied. Let $\dot{s}^*_\beta(s)$ denote the minimum time velocity profile which includes a stop at β, see Fig. 6.1. In the evaluation of the velocity profile $\dot{s}^*_\alpha(s)$ for an arbitrary $\alpha \in [0, \beta]$, the velocity upper bound $v^{\max}(\alpha)$ is used to guarantee that the robot can stop before entering the common workspace. The profile $\dot{s}^*_\beta(s)$ is thus used to determine $v^{\max}(\alpha)$ to fulfill this requirement. The maximal permitted velocity at the clearance point is obtained from the corresponding point on $\dot{s}^*_\beta(s)$. The constraint in (6.6) is therefore updated to

$$\dot{s}(\alpha) \leq \dot{s}^*_\beta(\alpha), \quad \alpha \in [0, \beta] \tag{6.13}$$

The velocity is at every path position less than, or equal to, the velocity of the unrestricted solution, i.e. $\dot{s}^*_\alpha(s) \leq \dot{s}^*_0(s), \forall s \in [0, 1], \alpha \in [0, \beta]$. Consider a point γ corresponding to the maximal path position where, for an arbitrary α, $\dot{s}^*_0(s)$ and $\dot{s}^*_\alpha(s)$ are identical.

Definition 6.2.1. Let γ be a point defined as

$$\gamma = \max \ s \quad \text{s.t. } \dot{s}^*_\alpha(\xi) = \dot{s}^*_0(\xi), \ \forall \xi \in [0, s] \tag{6.14}$$

for an arbitrary $\alpha \in [0, \beta]$. Hence, γ is the maximal path position for which $\dot{s}^*_\alpha(s)$ equals $\dot{s}^*_0(s)$ for all prior points along the path, see Fig. 6.1.

The time difference, denoted as ΔT, between executing a trajectory specified for a given α, i.e. $T^*(\alpha, 0)$, compared to the unrestricted solution, $T^*(0, 0)$, is given by

$$\Delta T(\alpha) = T^*(\alpha, 0) - T^*(0, 0) = \int_0^1 \frac{1}{\dot{s}^*_\alpha(s)} - \frac{1}{\dot{s}^*_0(s)} \, ds \tag{6.15}$$

In Paper 5, we show that $\Delta T(\alpha) = 0$ if $\alpha \leq \gamma$ and $\Delta T(\alpha) > 0$ if $\alpha > \gamma$. As illustrated in Fig. 6.1, the robot decelerates after reaching γ, since the velocity bound $v^{\max}(\alpha)$ becomes more restrictive as α approaches β, which explains why the time to traverse the path $\dot{s}^*_\alpha(s), \alpha > \gamma$ exceeds the time for the unrestricted solution.

So far, $\tau = 0$ has been considered, corresponding to a minimum final time given by $T^*(\alpha, 0)$. To incorporate τ to the final time expression, $T^*(\alpha, 0)$ is first divided into two parts. The first part considers the minimum time in which α is reached, while the second part relates to the minimum time to move from α to the end position. Let $t^*_\alpha(s)$ denote the minimum time in which a path position s can be reached when executing $\dot{s}^*_\alpha(s)$. Hence, the shortest time to reach α is expressed as $t^*_\alpha(\alpha)$. For notational convenience, this expression is reduced to t^*_α. The time to move from α to the end position is denoted as $\bar{t}^*_\alpha$, resulting in a final time given by

$$T^*(\alpha, 0) = t^*_\alpha + \bar{t}^*_\alpha \tag{6.16}$$

Due to the timing constraint in (6.7), the robot is not allowed to traverse α before time τ. Hence, when also considering τ, the minimum final time is expressed as

$$T^*(\alpha, \tau) = \max(t^*_\alpha, \tau) + \bar{t}^*_\alpha \tag{6.17}$$

where $\max(t^*_\alpha, \tau)$ corresponds to the time at which α is reached. For $\tau > t^*_\alpha$, $\dot{s}^*_\alpha(s)$ can have infinite number of solutions. For one of these solutions, the robot waits in the initial position until time $\tau - t^*_\alpha$, after which it moves to α in minimum time, arriving at time τ. Note that the corresponding velocity profile, when parametrized by position, is identical to the velocity profile for the case $\tau < t^*_\alpha$, when the robot starts immediately and executes the minimum time velocity profile. However, expressed in terms of time, i.e. $\dot{s}^*(t)$, the velocity profiles differ since for $\tau > t^*_\alpha$, the robot waits in its initial position. Another feasible solution when $\tau > t^*_\alpha$ is to start immediately and execute the path up to α with reduced velocity. This approach is studied in Section 6.4. Arriving early, implying that the common workspace is occupied, results in the robot coming to a stop at the boundary of the shared space. The impact on the final time for changes in α and τ is studied next, after which the possibility to reduce the energy consumption is examined.

6.3 Minimum Time Sensitivity

The final time in (6.17) is a function of both α and τ. To analyze how changes in these parameters impact the final time, the partial derivative of $T^*(\alpha, \tau)$ with respect to τ is studied,

$$\frac{\partial T^*(\alpha, \tau)}{\partial \tau} = \begin{cases} 0, & \text{if } \tau \leq t^*_\alpha \\ 1, & \text{else} \end{cases} \tag{6.18}$$

According to (6.18), the final time is not affected by changes in τ as long as $\tau \leq t^*_\alpha$. In this case, the minimum time in which the robot is able to reach α is greater than the time at which the common zone assumingly becomes free. The

robot will thus execute its time minimal trajectory to achieve the best possible final time. As τ increases and $\tau \leq t^*_\alpha$ still holds, the final time remains unaffected. While $\tau > t^*_\alpha$, the final time increases linearly as τ increases in (6.18), since in this case $T^*(\alpha, \tau) = \tau + \bar{t}^*_\alpha$. To analyze how changes in α impact the final time, we study the partial derivative of $T^*(\alpha, \tau)$ with respect to α,

$$\frac{\partial T^*(\alpha, \tau)}{\partial \alpha} = \begin{cases} t^{*\prime}_\alpha + \bar{t}^{*\prime}_\alpha, & \text{if } \tau \leq t^*_\alpha \\ \bar{t}^{*\prime}_\alpha, & \text{else} \end{cases} \tag{6.19}$$

where $(\,)' = d/d\alpha$. In Paper 5, we show that the time difference between executing $\dot{s}^*_\alpha(s)$ and $\dot{s}^*_0(s)$ is equal to 0 if $\alpha \leq \gamma$. Note that $\tau = 0$ was considered. However, the velocity profile is in this case equal to the velocity profile when $\tau \leq t^*_\alpha$ and the robot executes its path in shortest possible time. The partial derivative in (6.19) is thus equal to 0 if $\alpha \leq \gamma$ and $\tau \leq t^*_\alpha$, which gives

$$t^{*\prime}_\alpha = -\bar{t}^{*\prime}_\alpha, \quad \text{if } \alpha \leq \gamma,\ \tau \leq t^*_\alpha \tag{6.20}$$

The minimum time required to reach α is strictly increasing as α increases, i.e. $t^{*\prime}_\alpha > 0$. As a consequence, $\bar{t}^*_\alpha$ is strictly decreasing as α increases when $\alpha \leq \gamma$. The position of the clearance point can thus be changed without affecting T if $\alpha \leq \gamma$ and $\tau \leq t^*_\alpha$. Recall that $\Delta T(\alpha) > 0$ if $\alpha > \gamma$ which implies that (6.20) no longer holds. When $\tau > t^*_\alpha$, the change in final time equals $\bar{t}^{*\prime}_\alpha$. The time $\bar{t}^*_\alpha$ can be expressed as

$$\bar{t}^*_\alpha = \int_\alpha^1 \frac{1}{\dot{s}^*_\alpha(s)} ds = \int_\alpha^1 \frac{1}{\dot{s}^*_0(s)} ds, \quad \alpha \leq \gamma \tag{6.21}$$

since for $s \in [\alpha, 1]$, $\dot{s}^*_\alpha(s) = \dot{s}^*_0(s)$ if $\alpha < \gamma$. The derivative is given by

$$\bar{t}^{*\prime}_\alpha = -\frac{1}{\dot{s}^*_0(\alpha)} \leq 0, \quad \alpha \leq \gamma \tag{6.22}$$

which is based on the fact that $\dot{s}^*_0(\alpha) > 0,\ \forall \alpha \in \{0, \gamma\} \setminus 0$. As a result, the final time decreases as α increases when $\alpha \leq \gamma$ and $\tau > t^*_\alpha$.

The sign of $\bar{t}^{*\prime}_\alpha$ for $\alpha > \gamma$ cannot be established. In this case, an increase in α implies a decrease in $v^{\max}(\alpha)$. The robot will execute the path after α as fast as possible since minimum time is prioritized. An increase in α thus implies a reduction in the initial velocity for the path $s \in [\alpha, 1]$. A question left to be answered is if the additional distance traveled as α increases, results in a decrease in $\bar{t}^*_\alpha$, or if the reduced initial velocity for traversing the remaining path results in an increase in $\bar{t}^*_\alpha$.

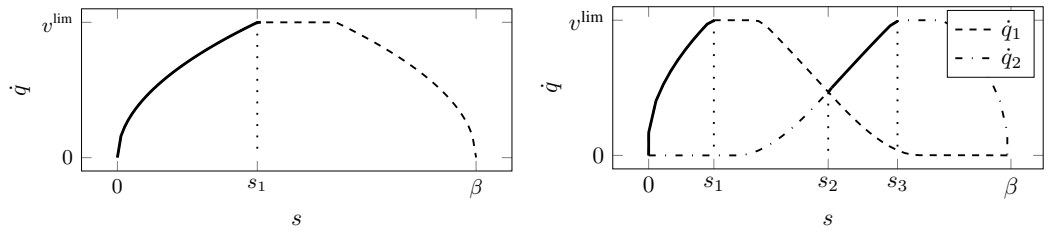
Based on the performed analysis, conclusions concerning the position of α, as well as the timing τ can be drawn. In (6.18) and (6.19), when $\tau \leq t^*_\alpha$ and $\alpha \leq \gamma$, the resulting solutions achieve a final time corresponding to the unrestricted solution. In this case, the common zone becomes free before the robot is able to reach it. For an arbitrary $\tau > t^*_\alpha$ when $\alpha \leq \gamma$, the final time decreases as α increases. Hence, the clearance point should be placed as far along the path as possible, which in this case equals $\alpha = \gamma$. We will later return to (6.19) and the behavior of $\bar{t}^{*\prime}_\alpha$ for $\alpha > \gamma$ in Section 6.5 to study the effect on the final time for increasing α. Next, energy reduction analysis is performed.

6.4 Energy Reduction

In this section, the impact on the energy consumption for different choices of τ and α is studied. In (Riazi et al. 2017), the results show that the energy consumption can be approximated as the squared joint acceleration. As a result, the system becomes more energy efficient by reducing the velocities and accelerations while moving along a path. A drawback is extended execution times which can affect the final time, regarded as the prioritized objective in this work. When $\tau > t^*_\alpha$, redundant time is available. The path $s \in [0, \alpha]$ can thus be executed with reduced velocities, without affecting the final time. Two terms indicating if energy reduction is feasible, *slack time* and *slack interval*, are presented. First, points where the robot switches to and from maximum acceleration are introduced.

Definition 6.4.1 (Switch Points)**.** Let $s_i,\ i \in [1, N]$ define N points where the robot switches to and from maximum acceleration. Let $s_0 = 0$ and $s_{N+1} = \beta$.

Note that the total number of switch points N is uneven, since the robot will start with an acceleration segment and end with a stop, see Fig. 5.2. If the robot can reach the clearance point before the common workspace becomes available, i.e. $\tau > t^*_\alpha$, the redundant time equals $\tau - t^*_\alpha$. The slack time is characterized in the following definition.



(a) The switch point s_1 for a robot with one joint.

(b) The switch points $s_i, i \in \{1, 2, 3\}$ for a robot with two joints, q_1 and q_2. The bound for both $\dot{q}_1(s)$ and $\dot{q}_2(s)$ is equal to $v^{\lim}$.

Figure 6.2: Minimum time velocity profiles for a robot with (a) one joint and (b) two joints specifying the switch points in the interval $s \in [0, \beta]$. The solid lines corresponds to the segments where maximum acceleration is maintained.

Definition 6.4.2 (Slack time)**.** Let t_E denote the slack time in the system, characterized as

$$t_E = \begin{cases} 0, & \text{if } \ \alpha < s_1 \text{ or} \\ & \quad \tau \leq t^*_\alpha \\ \tau - t^*_\alpha, & \text{else} \end{cases} \tag{6.23}$$

If α is located during the first acceleration phase, i.e. $\alpha < s_1$, the slack time is equal to 0 since maximum acceleration is required to satisfy $\dot{s}(\alpha) = v^{\max}(\alpha)$. Note

that, although redundant time is available, i.e. $\tau - t^*_\alpha > 0$, and $\alpha < s_1$, the slack time equals 0, which is also the case when redundant time is unavailable, i.e. $\tau \leq t^*_\alpha$. The slack interval is used to determine the stretch along the path, starting from the initial position, where slack time can be used to reduce the velocities for the robot without affecting the final time.

Definition 6.4.3 (Slack interval)**.** The slack interval is given by $[0, s_E]$, where s_E, is defined as

$$s_E = \left\{ \begin{array}{ll} s_i, & \text{if } i \text{ even}, \\ \alpha, & \text{if } i \text{ uneven}, \end{array} \right\} \begin{array}{l} s_i \leq \alpha \leq s_{i+1}, \\ i \in [0, N] \end{array} \tag{6.24}$$

If $\alpha \in [s_0, s_1]$, which represents the first acceleration segment, the slack interval equals $[0, s_0 = 0]$, implying that no stretch is available for energy reduction. For a robot with one joint, the slack interval equals $[0, \alpha]$ when $\alpha > s_1$. Furthermore, if $t_E > 0$, the robot can use the redundant time $\tau - t^*_\alpha$ and execute the path up to α with reduced velocities, as illustrated in Fig. 6.3. Note that $\dot{s}^*_\alpha(\alpha) = \dot{s}^*_\beta(\alpha)$, which shows that the maximum possible velocity, guaranteeing the possibility to stop at β, is attained at the clearance point. A remark has to be made regarding $\alpha > \gamma$. Since this implies that $\dot{s}^*_\alpha(\alpha) \leq \dot{s}^*_0(\alpha)$ and the path $s \in [\alpha, 1]$ is executed in minimum time, the robot will accelerate as soon as α is traversed. As a result, the energy consumption increases due to the correlation between acceleration and energy consumption previously mentioned.

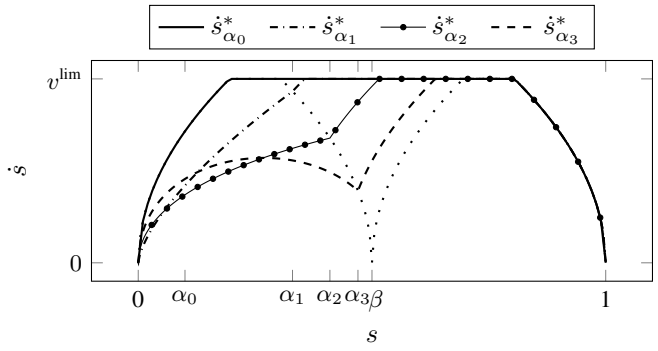


Figure 6.3: Velocity profiles $\dot{s}^*_\alpha$, $\alpha \in \{\alpha_0, \alpha_1, \alpha_2, \alpha_3\}$ for $\tau \geq t^*_\alpha$. When $t_E > 0$, the robot can traverse the path with reduced velocities if $\alpha > s_1$. Note that for $\alpha_0 < s_1$, this is not possible. The dotted line corresponds to $\dot{s}^*_\beta(s)$.

A velocity profile considering a robot with two joints is depicted in Fig. 5.2b. Initially, one of the joints has 0 velocity. At some point, both joints are in motion and finally, the joint with the later start is solely in motion. The number of switch points increases in this case. Note, if $\alpha \in [s_2, s_3]$, the robot has to achieve the maximum possible velocity at s_2 in order to complete the path after α in minimum time, i.e. to be able to reach $\dot{s}^*_\alpha(\alpha)$ then $\dot{s}(s_2) = \dot{s}^*_\alpha(s_2)$. In Fig. 6.4, s_E is depicted together with the switch points for a robot with two joints. Based on Definition 6.4.3, s_E corresponds to the length of the slack interval. If the acceleration segments are

disregarded, an increase in α corresponds to an increase in slack interval. Hence, the further along the path the clearance point is located, the longer the stretch is where eventual slack time can be distributed.

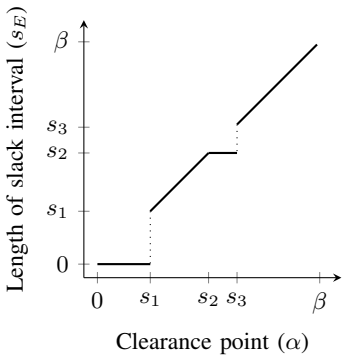


Figure 6.4: The slack position s_E for the robot with two joints where $s_i,\ i \in \{1, 2, 3\}$, are switch points. As long as α is not located during the acceleration segments, i.e. $\alpha \notin [0, s_1]$ and $\alpha \notin [s_2, s_3]$, the slack position increases as α increases.

Based on the performed analysis, slack time is a key factor when energy consumption reduction is feasible. Another requirement considers the placement of α, where the initial acceleration phase should be avoided in order to reduce the energy consumption. As α approaches β, the slack interval increases. On the downside, when $\alpha > \gamma$, an increase in α implies that the acceleration required to complete the remaining path in minimum time increases, which has a negative impact on energy efficiency.

6.5 Example

The impact on the final time and energy consumption for different combinations of α and τ is illustrated in this section by introducing an example, where a robot with one joint is analyzed. The robot moves along a path given by $q(s) = s, s \in [0, 1]$. The workspace for the robot overlaps with the workspace of another robot, assumed to have higher priority in terms of the order in which the common workspace is occupied. The boundary of the common workspace is located at the middle of the path, i.e. $\beta = 0.5$. The clearance point is hence restricted to $0 \leq \alpha \leq 0.5$. The velocity and acceleration limits for the joint are $v^{\text{lim}}(s) = 1.25$ and $a^{\text{lim}} = 4$, whereas the energy consumption E is estimated by

$$E = \int_0^1 \ddot{\mathbf{q}}(s)^T \ddot{\mathbf{q}}(s) ds \tag{6.25}$$

To study how the timing and position related to the clearance point impact the performance, a weighted sum of objectives prioritizing the final time T is considered when solving the mathematical model in (6.12). The results are discussed and compared to the conclusions made in Section 6.3 and Section 6.4.

6.5.1 Final Time

The final time for different values of α and τ is depicted in Fig. 6.5. Based on the analysis performed in Section 6.3, the effect on the final time will differ depending on whether τ is less than, or greater than, t^*_α as well as the position of α. Next, results obtained from Fig. 6.5, are analyzed by studying Regions I-IV individually.

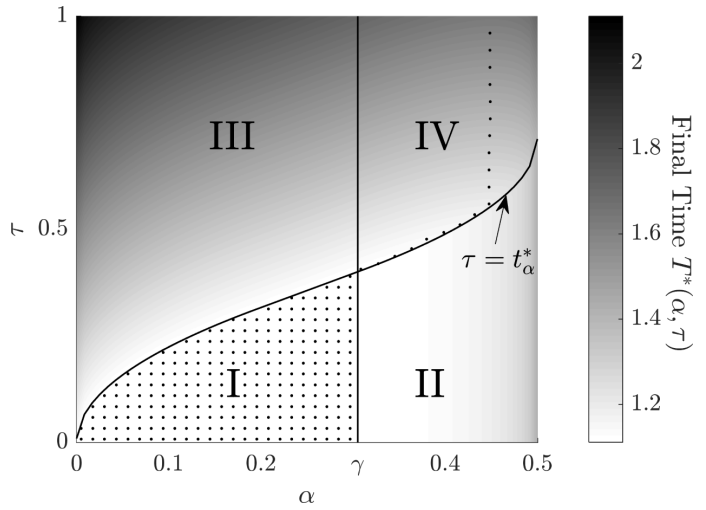


Figure 6.5: The final time, $T^*(\alpha, \tau)$, for different combinations of α and τ. The dots indicate the time minimum choice of α, given a certain τ. In region I and II, $\tau \leq t^*_\alpha$ whereas in region III and IV, $\tau > t^*_\alpha$. For a given $\tau \leq t^*_\alpha$, the optima is located in region I where $\alpha \leq \gamma$. For $\tau > t^*_\alpha$, a unique optimum is obtained along the dotted line in region IV.

Region I

In this region, where $\tau \leq t^*_\alpha$ and $\alpha \leq \gamma$, the final time T is constant. Based on (6.18), this result is expected since when the robot is unable to reach α before τ, the robot will execute its minimum time velocity profile. Hence, as τ increases and $\tau \leq t^*_\alpha$, T is not affected since the robot can not traverse the path faster than its minimum time trajectory. According to (6.19) and (6.20), the final time is constant for changes in α. However, when $\alpha > \gamma$, corresponding to region II, this no longer holds.

Region II

The clearance point is located along the deceleration phase in this region, i.e. $\alpha > \gamma$, and $\tau \leq t^*_\alpha$. In region II, T increases as α increases. This is due to the behavior in Fig. 6.6, where $t^{*'}_\alpha \geq -\bar{t}^{*'}_\alpha$ for $\alpha > \gamma$ resulting in a larger T according to (6.19).

Region III

In this region, $\tau > t^*_\alpha$ and $\alpha \leq \gamma$. As illustrated in Fig. 6.6 and stated in (6.18), T increases as τ increases. In (6.19), the change in final time as α increases equals $\bar{t}^{*'}_\alpha$ in this region. Since $\bar{t}^{*'}_\alpha$ decreases as $\alpha \leq \gamma$ increases, T decreases, see Fig. 6.6.

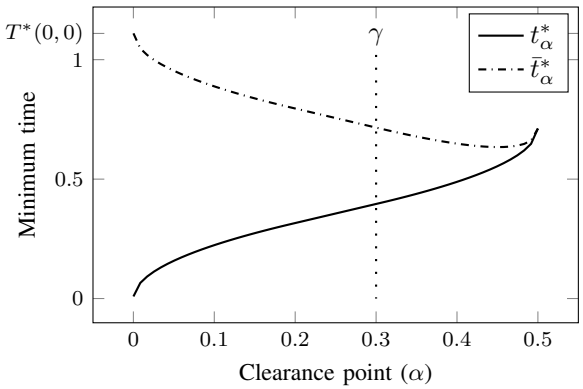


Figure 6.6: The minimum time in which α can be reached, t^*_α, and the minimum time to complete the remaining path after α, $\bar{t}^*_\alpha$. For $\alpha < \gamma$, it can be noticed that the curves are symmetric such that $t^{*'}_\alpha = -\bar{t}^{*'}_\alpha$. For $\alpha = 0$, $t^*_\alpha = 0$ whereas $\bar{t}^*_\alpha$ equals the final time of the unrestricted solution, $T^*(0,0)$.

Region IV

The change in final time as τ increases is equal to the behavior in region III. In Section 6.3, a question was raised regarding the impact on T when $\alpha > \gamma$ increases and $\tau > t^*_\alpha$, i.e. region IV. As for region III, the partial derivative in (6.19) is equal to $\bar{t}^{*'}_\alpha$. However, in this region, where $\alpha > \gamma$, $\bar{t}^*_\alpha$ reaches a minimum when α approaches β, after which it increases. Hence, the same behavior applies for T. For a given τ, the optimal clearance point is obtained by choosing an α such that the solution $T(\alpha, \tau)$ coincides with the dotted line in region IV, depicted in Fig. 6.5.

Based on the presented results, the final time is constant in region I. Hence, for $\tau < t^*_\alpha$, α can in this case be arbitrarily positioned along the path up to $s = \gamma$ without impacting the final time. In Section 6.3, we concluded that for an arbitrary $\tau > t^*_\alpha$, the position of the clearance point should be located at $\alpha = \gamma$, since the behavior of $\bar{t}^*_\alpha$ nor its derivative could not be established for $\alpha > \gamma$. Based on this example, the results show that for a 1-joint robot, the optimal placement of α for a given τ, is in region IV where $\alpha > \gamma$. This results in a clearance point located along the deceleration segment for the robot. However, if α is placed too close to β, the time to traverse the remaining path increases resulting in an increase in final time.

6.5.2 Energy Consumption

In Fig. 6.7, the energy consumption for different combinations of α and τ is depicted. As before, different regions are introduced representing the cases when $\tau \leq t^*_\alpha$ and $\tau > t^*_\alpha$. According to Fig. 5.2a, a robot with one joint has one switch point s_1. The regions are thus determined based on whether $\alpha \leq s_1$, $s_1 < \alpha \leq \gamma$ or $\alpha > \gamma$. This results in six regions where the energy consumption is studied, as well as the optimal location of α for a given τ.

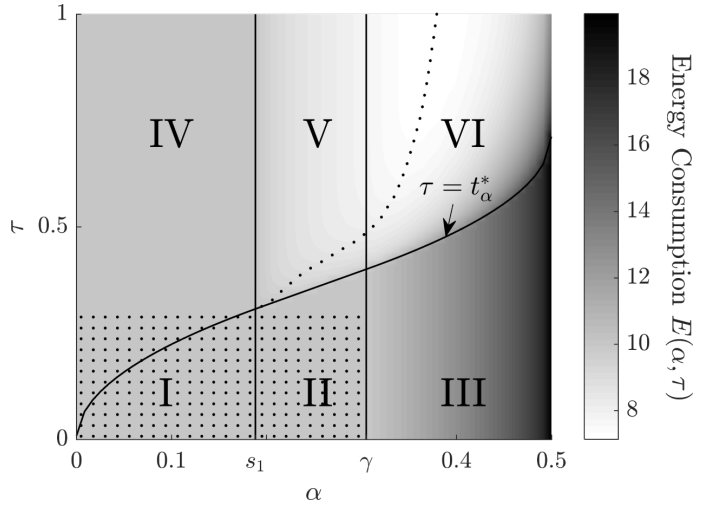


Figure 6.7: The energy consumption, $E(\alpha, \tau)$, for different combinations of α and τ. The dots indicate the energy minimum choice of α, given a certain τ. Regions I-VI distinguish between the cases when $\tau \leq t^*_\alpha$ and $\tau > t^*_\alpha$, as well as whether $\alpha \leq s_1$, $s_1 < \alpha \leq \gamma$ or $\alpha > \gamma$. For small τ, the optima is obtained in the region where $\alpha \leq \gamma$. As τ increases, a unique optimum is obtained along the dotted line in region V ($s_1 \leq \alpha \leq \gamma$). For even larger τ, the optimum is located in region VI ($\alpha > \gamma$).

Region I and II

In these regions, $\tau \leq t^*_\alpha$. Moreover, $\alpha \leq s_1$ in Region I, while $s_1 \leq \alpha \leq \gamma$ in Region II. For both cases, $\dot{s}^*_\alpha(s) = \dot{s}^*_0(s)$ since the common workspace becomes available before the robot is able to reach it. Based on (6.23), $t_E = 0$ when $\tau < t^*_\alpha$ and a reduction in E is not achievable. Note that the energy consumption is equal and constant in these regions since the velocity profiles are identical.

Region III

Since $\tau \leq t^*_\alpha$ and $\alpha > \gamma$ in this region, the velocity achieved at α is less than the maximal velocity, i.e. $\dot{s}^*_\alpha(\alpha) < \dot{s}^*_0(\alpha)$. Since the remaining path after α is executed in minimum time, implying α is followed by an acceleration phase, an increase in the energy consumption is expected. Hence, the closer to β that α is positioned, the more E increases. For a given α, E is constant as τ increases as long as $\tau \leq t^*_\alpha$ still holds.

Region IV

The slack time, enabling energy reduction, is non-existing in the regions examined so far. Even though $\tau > t^*_\alpha$ in this region, E is not reducible since $\alpha \leq s_1$, and hence $t_E = 0$. The energy consumption is equal to the one in Regions I and II, implying that the velocity profiles are identical. The robot waits in its initial position, after which the unrestricted velocity profile is executed such that α is reached at time τ, thus consuming the same amount of energy. For a certain τ in the interval

represented by the dotted area in Fig. 6.7, the energy minimal solution is obtained by positioning the clearance point anywhere in between $0 \leq \alpha \leq \gamma$. For greater τ, the optimum is found in the following regions.

Region V

In this region, where $\tau > t^*_\alpha$ and $s_1 \leq \alpha \leq \gamma$, there is redundant time available in the system. Hence, a reduction in E is possible since the robot can execute its path up to α with reduced velocities and accelerations. As a consequence, E decreases as α increases, see Fig. 6.7. The dotted line shows that as τ increases, the optimal solution is obtained by placing α further along the path.

Note that, for a certain α, when τ increases beyond the value of the optimal solution represented by the dotted line, E is constant. Even though slack time is available, E cannot be further reduced. A possible explanation is that after traversing α, the robot should execute the remaining path in minimum time. As a consequence, the robot achieves its maximal possible velocity at α, i.e. $v^{\max}(\alpha)$. In order to obtained a more energy efficient solution, a reduction in velocities along the stretch $s \in [0, \alpha]$ is necessary. However, this is only possible as long as $v^{\max}(\alpha)$ is still achievable at α. Hence, after a certain increase in τ, the energy consumption becomes constant. There is a thus a trade-off between energy consumption versus minimum time.

Region VI

As illustrated in Region VI in Fig. 6.7, the energy consumption decreases for a certain τ when $\alpha > \gamma$ increases. Even though the cost for accelerating after α is introduced, the most energy efficient solutions are obtained in this region. Due to the increase in slack time since $\tau > t^*_\alpha$, as well as the increase in slack position resulting from the increase in α, the cost for accelerating is compensated for. Note that E increases as α approaches β. The dots depicts the optimal clearance point positions as τ increases. If a friction cost and/or gravitation cost are added to the objective, E is expected to eventually increase as τ increases.

To summarize the results on energy reduction, one necessary requirement to retrieve an energy efficient solution is to have slack time, i.e. that $t_E > 0$. If this condition is fulfilled and α is placed along the stretch $s_1 \leq \alpha \leq \gamma$, this results in reduced energy consumption. When $\alpha > \gamma$, a further decrease in E is possible. However, a trade-off exists between increasing the slack position and the acceleration required to complete the remaining path after α in minimum time.

6.6 Summary

In this chapter, an approach to embed robustness into trajectory planning was presented. Influenced by an industrial approach, a clearance point α was introduced, where the availability of a common workspace was evaluated. The velocity is restricted

at the this location, guaranteeing that the robot can stop before entering the common workspace in case it is not yet available due to an uncertainty. The time τ at which the shared space is expected to become available is assumed to be known. The impact on the performance for changes in τ and α is studied. The results show the optimal placement of α given a certain τ in terms of the final time and energy consumption, when minimum time is prioritized.

Chapter 7

Summary of Appended Papers

This chapter summarizes the appended papers in Part II. The contributions are highlighted and the relation between papers is briefly discussed.

Paper 1

Nina Sundström, Oskar Wigström, Petter Falkman and Bengt Lennartson. "Optimization of Operation Sequences using Constraint Programming". *Proceedings of the 14th IFAC Symposium on Information Control Problems in Manufacturing, 45 (6), 1580-1585, 2012.*

This paper presents a method to connect the design and optimization of a production system. The modeling of operation sequences, corresponding to the considered production system, is performed in SP. The operation model is transformed to a CP optimization model. An abstraction method, called work equivalence, is introduced to enable alternative model formulations. A case study of an aero engine assembly structure plant is presented, where the efficiency of the different formulations is evaluated. The performance of the abstraction formulations exceeds the performance of a standard formulation.

Paper 2

Nina Sundström and Bengt Lennartson. "Event- and Time-Based Design of Operation Sequences with Uncertainties in Execution Times". *Proceedings of the IEEE 18th Conference on Emerging Technologies Factory Automation (ETFA), 2013.*

In this paper, the approach in Paper 1 is extended. The aim is to close the loop by retrieving the results from the optimization of a production system modeled in SP. More specifically, conditions expressing when operations can start based on the sequences in the time-optimal schedule is added to the original operation model in SP. Hence, the time-based solution becomes an event-based description, ensuring that the sequences in the schedule are maintained. If uncertainties are present, unnecessary delays are avoided by examining, and possibly relaxing, the logical restrictions added in the first step.

Paper 3

Nina Sundström and Bengt Lennartson. "Rescheduling Affected Operations - a Purely Predictive Approach". *Proceedings of the 13th International Workshop on Discrete Event Systems (WODES), 2016.*

The rescheduling framework and two underlying methods are studied in this paper. AOR is a method used in the literature on rescheduling with the purpose to repair schedules if disruptions are present. Only operations directly or indirectly affected by a disruption are rescheduled. The approach is based on a deterministic, minimum time schedule for a job shop generated offline which is used online to update the schedule in response to machine breakdowns. This paper presents a purely offline approach to retrieve the same behavior and results. The proposed procedure is based on scheduling results using a disjunctive graph. Any type of disruption causing delays can be considered. The paper also includes RSR which is a method where all remaining operations are postponed if a delay occurs. This conservative method is a common benchmark in performance analysis. A formal proof is presented to show that AOR always performs better than, or equal to, RSR if delays are present.

Prior to performing feasibility tests, the proposed method in Paper 2 will, if disruptions occur, delay the start of all succeeding operations, such that the order in which they start resembles the order in the time-minimal schedule. This behavior is similar to RSR. If logical restrictions can be relaxed as a consequence of performing feasibility tests, the solution thus becomes less restrictive. However, in comparison to Affected Operation Rescheduling, the solutions based on the approach in Paper 2 are more restrictive. In Paper 1 and Paper 2 general shop floors are considered, whereas the approach presented in this paper considers the JSP.

Paper 4

Nina Sundström, Oskar Wigström and Bengt Lennartson. "Conflict Between Energy, Stability, and Robustness in Production Schedules". *IEEE Transactions on Automation Science and Engineering, 14 (2), 658-668, 2017.*

Due to methods considered in rescheduling where idle time can be inserted into a schedule to protect it against disruptions, a possible conflict arise when rescheduling techniques are combined with energy optimization. In energy optimization, results show that idle time in a schedule can be used to extend the duration of an operation and, as a result, reduce the energy consumption. In this paper, a systematic approach to evaluate this conflict is proposed.

In rescheduling, robustness and stability are terms used to evaluate the performance of a schedule. Existing surrogate measures of robustness and stability are presented as well as a proposed stability measure. The performance of the measures are evaluated. An energy consumption signature for an operation is derived from experiments on an industrial robot, resulting in a convex model parameterized by processing time.

To analyze the trade-off, surrogate measures of robustness and stability are used together with makespan and energy consumption in a multi-objective MINLP

formulation. The results show that an increase in energy efficiency results in a system less sensitive to disruptions. Also, a conflict between stability and robustness becomes apparent.

Paper 5

Nina Sundström, Oskar Wigström and Bengt Lennartson. "Robust and Energy Efficient Trajectories in a Common Workspace Setting". *Submitted for possible journal publication, 2017.*

A trajectory planning approach is presented in this paper, where robustness is embedded to, under all circumstances, guarantee collision-free scenarios. A clearance point is introduced along the path for a robot where a decision is taken to either enter a shared zone, or stop at the boundary of it. This puts a restriction on the velocity at the clearance point. The closer to the boundary of the shared space the clearance point is located, the more restricted the velocity is.

A problem formulation is stated, assuming a predefined path including robot dynamics and robust constraints with multiple objectives, to minimize final time and energy consumption. The impact on the performance concerning the timing and position related to the clearance point is analyzed. Based on the time at which the common workspace is expected to become available, the optimal clearance point location is obtained.

Chapter 8

Concluding Remarks

Four research questions, RQ1-RQ4, were stated in Section 1.2, which have been addressed throughout this thesis. The overall topic related to these questions concerns scheduling of production systems, with emphasis on robustness and energy efficiency. This chapter will conclude the presented work and discuss directions of future work.

The goals when developing schedules for production systems are manifold. However, systems do not always progress as intended due to unforeseen disruptions, which can result in out-of-date schedules and goals not met. To cope with disturbances, methods to develop robust and stable schedules are presented in this thesis. In literature, different measures have been presented to evaluate the quality of a schedule. This work considers delay in makespan between the obtained schedule and the run-time schedule as a robustness measure, whereas stability is measured in terms of start time deviations or sequence deviation.

The production systems regarded in this thesis are modeled by either using the SP model or the disjunctive graph. To address RQ1 and RQ2, an approach is proposed in Paper 1 where the SP model is converted to a CP model to generate a time-optimal schedule. The precedence relations, based on the execution order in the schedule, are formulated as event-based conditions in the SP model, which preserves the order if delays are present. The resulting schedule is hence stable with respect to sequence deviations. Furthermore, the method attempts to relax conditions by examining logical restrictions between tasks, with the intention to reduce the effect of disruptions. The stated procedure is presented in Paper 2. The disjunctive graph is used to show that an already established offline and online method, called AOR, that generates stable schedules with maintained sequences, can be performed completely offline. This work is presented in Paper 3 to give an answer to RQ1 and RQ2.

In minimum time schedules, idle time is a result of tasks sharing resources and consequently have to wait for a resource to become available. If a task which is succeeded by slack is delayed, the schedule is unaffected. Approaches where additional slack is embedded into schedules are thus used in the literature to develop robust and stable schedules, often by considering surrogate measures of robustness and stability that includes slack. A method, proposed to answer RQ2 and RQ3, is introduced in Paper 4, where surrogate measures are included as objectives into a problem formulation to generate robust and stable schedules. A convex stability measure is

proposed with the goal to distribute slack such that the impact of delays with known probability density functions can be minimized in terms of start time deviations.

Result in energy optimization literature show that energy consumption can be decreased by reducing the acceleration for a robot, implying an increase in execution time. Available slack can thus be diminished on behalf of reduced energy consumption. A problem formulation is stated with RQ3 in mind, taken both makespan, energy consumption, stability and robustness into consideration to show the conflict that arises and the trade-off that has to be made when developing both robust and energy efficient schedules. The results are presented in Paper 4. A convex energy signature parameterized by execution time is used to measure the energy consumption with the assumption that the nominal execution time is lower than that achieving minimal energy consumption.

A robust trajectory planning approach is also suggested in this thesis, attempting to answer RQ4. The path as well as the order in which two robots use a common workspace are predefined. The velocity is bounded at a certain position along the path, called the clearance point, where the availability of the shared space is evaluated. If occupied, the robot has to stop at the boundary of the common space. Otherwise, the robot can continue on its path and enter the shared space. By studying the impact on the final time and energy consumption for different positions and time at which this point is traversed, the optimal clearance point is presented for different amount of slack. The method and results are given in Paper 5.

An interesting extension would be to include multiple points where the availability is evaluated. Another approach could be to regard the time at which the common workspace becomes available as stochastic with a probability density function to analyze the effect on the performance. The order in which the robots use the common workspace is assumed to be known in this work. A possible continuation could therefore be to include this decision into the problem formulation.

Future work, combining robustness and energy efficiency, can regard resources with a speed scaling functionality. If disturbances are present, the effect of delays in final time can be reduced, or possibly eliminated, by executing tasks faster. A necessary prerequisite is the existence of redundant time. If the minimum time in e.g. workstations multiple robots, is less than the cycle time, this approach would result in an energy-efficient system with the capability to compensate for delays. Challenges arise concerning e.g. collision avoidance, which have to be tackled.

Bibliography

Abumaizar, R. J. and J. A. Svestka (1997). "Rescheduling job shops under random disruptions". In: *International Journal of Production Research* 35.7, pp. 2065–2082 (cit. on pp. 20, 21).

Akturk, M. S. and E. Gorgulu (1999). "Match-up scheduling under a machine breakdown". In: *European journal of operational research* 112.1, pp. 81–97 (cit. on p. 20).

An, Y.-J., Y. Kim, B. Jeong, and S.-D. Kim (2012). "Scheduling healthcare services in a home healthcare system". In: *Journal of the Operational Research Society* 63.11, pp. 1589–1599 (cit. on p. 4).

Bean, J. C., J. R. Birge, J. Mittenthal, and C. E. Noon (1991). "Matchup scheduling with multiple resources, release dates and disruptions". In: *Operations Research* 39.3, pp. 470–483 (cit. on p. 20).

Beck, J. C. and N. Wilson (2007). "Proactive algorithms for job shop scheduling with probabilistic durations". In: *Journal of Artificial Intelligence Research*, pp. 183–232 (cit. on p. 20).

Blażewicz, J., E. Pesch, and M. Sterna (2000). "The disjunctive graph machine representation of the job shop scheduling problem". In: *European Journal of Operational Research* 127.2, pp. 317–331 (cit. on p. 22).

Bobrow, J. E., S. Dubowsky, and J. Gibson (1985). "Time-optimal control of robotic manipulators along specified paths". In: *The international journal of robotics research* 4.3, pp. 3–17 (cit. on pp. 6, 43).

Bruzzone, A., D. Anghinolfi, M. Paolucci, and F. Tonelli (2012). "Energy-aware scheduling for improving manufacturing process sustainability: a mathematical model for flexible flow shops". In: *CIRP Annals-Manufacturing Technology* 61.1, pp. 459–462 (cit. on p. 35).

Cesta, A., A. Oddi, N. Policella, and S. F. Smith (2015). "A Precedence Constraint Posting Approach". In: *Handbook on Project Management and Scheduling Vol. 1.* Springer, pp. 113–133 (cit. on p. 13).

Chiang, T.-C. and L.-C. Fu (2007). "Using dispatching rules for job shop scheduling with due date-based objectives". In: *International journal of production research* 45.14, pp. 3245–3262 (cit. on p. 21).

Dai, M., D. Tang, A. Giret, M. A. Salido, and W. D. Li (2013). "Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm". In: *Robotics and Computer-Integrated Manufacturing* 29.5, pp. 418–429 (cit. on p. 5).

Daniels, R. L. and P. Kouvelis (1995). "Robust scheduling to hedge against processing time uncertainty in single-stage production". In: *Management Science* 41.2, pp. 363–376 (cit. on p. 20).

Davenport, A., C. Gefflot, and C. Beck (2014). "Slack-based techniques for robust schedules". In: *Sixth European Conference on Planning* (cit. on p. 5).

Dietmair, A. and A. Verl (2009). "A generic energy consumption model for decision making and energy efficiency optimisation in manufacturing". In: *International Journal of Sustainable Engineering* 2.2, pp. 123–133 (cit. on p. 5).

Diwekar, U. (2008). "Optimization under uncertainty". In: *Introduction to Applied Optimization*. Springer, pp. 1–54 (cit. on p. 20).

Gantt, H. L. (1903). "A graphical daily balance in manufacture". In: *Transactions of the American Society of Mechanical Engineers* 24, pp. 1322–1336 (cit. on p. 4).

Goren, S. and I. Sabuncuoglu (2008). "Robustness and stability measures for scheduling: single-machine environment". In: *IIE Transactions* 40.1, pp. 66–83 (cit. on pp. 5, 25).

Graham, R. L., E. L. Lawler, J. K. Lenstra, and A. R. Kan (1979). "Optimization and approximation in deterministic sequencing and scheduling: a survey". In: *Annals of discrete mathematics* 5, pp. 287–326 (cit. on p. 4).

Hazır, ö., M. Haouari, and E. Erel (2010). "Robust scheduling and robustness measures for the discrete time/cost trade-off problem". In: *European Journal of Operational Research* 207.2, pp. 633–643 (cit. on pp. 20, 26, 27).

He, W. and D.-h. Sun (2013). "Scheduling flexible job shop problem subject to machine breakdown with route changing and right-shift strategies". In: *The International Journal of Advanced Manufacturing Technology* 66.1-4, pp. 501–514 (cit. on p. 22).

Herrmann, J. (2006). *Handbook of Production Scheduling*. International Series in Operations Research & Management Science. Springer. ISBN: 9780387331171 (cit. on p. 9).

Al-Hinai, N. and T. ElMekkawy (2011). "Robust and stable flexible job shop scheduling with random machine breakdowns using a hybrid genetic algorithm". In: *International Journal of Production Economics* 132.2, pp. 279–291 (cit. on p. 26).

Hooker, J. (2000). *Logic-based methods for optimization: combining optimization and constraint satisfaction*. Wiley-Interscience series in discrete mathematics and optimization. John Wiley & Sons. ISBN: 9780471385219 (cit. on p. 12).

Hooker, J. (2005). "Planning and Scheduling to Minimize Tardiness". In: *Principles and Practice of Constraint Programming - CP 2005*. Springer Berlin Heidelberg (cit. on p. 9).

Jain, A. S. and S. Meeran (1999). "Deterministic job-shop scheduling: Past, present and future". In: *European journal of operational research* 113.2, pp. 390–434 (cit. on p. 22).

Jayamohan, M. and C. Rajendran (2000). "New dispatching rules for shop scheduling: a step forward". In: *International Journal of Production Research* 38.3, pp. 563–586 (cit. on p. 21).

Jensen, M. T. (2003). "Generating robust and flexible job shop schedules using genetic algorithms". In: *Evolutionary Computation, IEEE Transactions on* 7.3, pp. 275–288 (cit. on p. 26).

Johnson, S. M. (1954). "Optimal two-and three-stage production schedules with setup times included". In: *Naval Research Logistics (NRL)* 1.1, pp. 61–68 (cit. on p. 4).

Jorge Leon, V., S. David Wu, and R. H. Storer (1994). "Robustness measures and robust scheduling for job shops". In: *IIE transactions* 26.5, pp. 32–43 (cit. on pp. 5, 20, 25, 26).

Kan, A. R. (2012). *Machine scheduling problems: classification, complexity and computations*. Springer Science & Business Media (cit. on p. 22).

Katragjini, K., E. Vallada, and R. Ruiz (2013). "Flow shop rescheduling under different types of disruption". In: *International Journal of Production Research* 51.3, pp. 780–797 (cit. on p. 5).

Kerzner, H. (2013). *Project management: a systems approach to planning, scheduling, and controlling*. John Wiley & Sons (cit. on pp. 4, 10).

Kwok, Y.-K. and I. Ahmad (1999). "Static scheduling algorithms for allocating directed task graphs to multiprocessors". In: *ACM Computing Surveys (CSUR)* 31.4, pp. 406–471 (cit. on p. 4).

Lamas, P. and E. Demeulemeester (2015). "A purely proactive scheduling procedure for the resource-constrained project scheduling problem with stochastic activity durations". In: *Journal of Scheduling*, pp. 1–20 (cit. on pp. 13, 20).

Lambrechts, O., E. Demeulemeester, and W. Herroelen (2008). "A tabu search procedure for developing robust predictive project schedules". In: *International Journal of Production Economics* 111.2, pp. 493–508 (cit. on p. 20).

Lambrechts, O., E. Demeulemeester, and W. Herroelen (2011). "Time slack-based techniques for robust project scheduling subject to resource uncertainty". In: *Annals of Operations Research* 186.1, pp. 443–464 (cit. on pp. 5, 26).

LaValle, S. M. and S. A. Hutchinson (1998). "Optimal motion planning for multiple robots having independent goals". In: *IEEE Transactions on Robotics and Automation* 14.6, pp. 912–925 (cit. on p. 6).

Lennartson, B., K. Bengtsson, C. Yuan, K. Andersson, M. Fabian, P. Falkman, and K. Åkesson (2010). "Sequence planning for integrated product, process and automation design". In: *IEEE Transactions on Automation Science and Engineering* 7, pp. 791–802 (cit. on pp. 9, 10).

Levin, R. and C. Kirkpatrick (1966). *Planning and control with PERT/CPM*. New York, McGraw-Hill (cit. on p. 10).

Li, R.-K., Y.-T. Shyu, and S. Adiga (1993). "A heuristic rescheduling algorithm for computer-based production scheduling systems". In: *The International Journal Of Production Research* 31.8, pp. 1815–1826 (cit. on p. 20).

Liu, F., A. Narayanan, and Q. Bai (2000). "Real-time systems". In: (cit. on p. 6).

Lombardi, M., M. Milano, and L. Benini (2013). "Robust scheduling of task graphs under execution time uncertainty". In: *Computers, IEEE Transactions on* 62.1, pp. 98–111 (cit. on p. 13).

Lou, P., Q. Liu, Z. Zhou, H. Wang, and S. X. Sun (2012). "Multi-agent-based proactive–reactive scheduling for a job shop". In: *The International Journal of Advanced Manufacturing Technology* 59.1-4, pp. 311–324 (cit. on p. 20).

Mehta, S. V. and R. M. Uzsoy (1998). "Predictable scheduling of a job shop subject to breakdowns". In: *Robotics and Automation, IEEE Transactions on* 14.3, pp. 365–378 (cit. on p. 20).

Miettinen, K. (2012). *Nonlinear multiobjective optimization*. Vol. 12. Springer Science & Business Media (cit. on p. 38).

O'Donovan, R., R. Uzsoy, and K. N. McKay (1999). "Predictable scheduling of a single machine with breakdowns and sensitive jobs". In: *International Journal of Production Research* 37.18, pp. 4217–4233 (cit. on p. 22).

Papadakos, N. (2009). "Integrated airline scheduling". In: *Computers & Operations Research* 36.1, pp. 176–195 (cit. on p. 4).

Peng, J. and S. Akella (2005). "Coordinating multiple robots with kinodynamic constraints along specified paths". In: *The International Journal of Robotics Research* 24.4, pp. 295–310 (cit. on p. 6).

Pinedo, M. (2005). *Planning and scheduling in manufacturing and services*. Vol. 24. Springer (cit. on pp. 4, 9, 15).

Pinedo, M. (2012). *Scheduling: theory, algorithms, and systems*. Springer Science & Business Media (cit. on p. 19).

Raheja, A. and V. Subramaniam (2002). "Reactive recovery of job shop schedules–a review". In: *The International Journal of Advanced Manufacturing Technology* 19.10, pp. 756–763 (cit. on p. 20).

Riazi, S., O. Wigström, K. Bengtsson, and B. Lennartson (2017). "Energy and Peak Power Optimization of Time-Bounded Robot Trajectories". In: *IEEE Transactions on Automation Science and Engineering* 14.2, pp. 646–657 (cit. on pp. 5, 35, 47).

Rossi, F., P. v. Beek, and T. Walsh, eds. (2006). *Handbook of Constraint Programming (Foundations of Artificial Intelligence)*. New York, NY, USA: Elsevier Science Inc. (cit. on p. 12).

Roy, B. and B. Sussmann (1964). "Les problemes d'ordonnancement avec contraintes disjonctives". In: *Note ds* 9 (cit. on p. 22).

Sabuncuoglu, I. and S. Goren (2009). "Hedging production schedules against uncertainty in manufacturing environment with a review of robustness and stability research". In: *International Journal of Computer Integrated Manufacturing* 22.2, pp. 138–157 (cit. on pp. 5, 19, 26).

Sahinidis, N. V. (2004). "Optimization under uncertainty: state-of-the-art and opportunities". In: *Computers & Chemical Engineering* 28.6, pp. 971–983 (cit. on p. 20).

Salido, M. A., J. Escamilla, F. Barber, A. Giret, D. Tang, and M. Dai (2015). "Energy efficiency, robustness, and makespan optimality in job-shop scheduling problems". In: *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, pp. 1–13 (cit. on p. 5).

Silver, E. A., D. F. Pyke, R. Peterson, et al. (1998). *Inventory management and production planning and scheduling*. Vol. 3. Wiley New York (cit. on p. 4).

Siméon, T., S. Leroy, and J.-P. Lauumond (2002). "Path coordination for multiple mobile robots: A resolution-complete algorithm". In: *IEEE Transactions on Robotics and Automation* 18.1, pp. 42–49 (cit. on p. 6).

Subramaniam, V. and A. Raheja (2003). "mAOR: A heuristic-based reactive repair mechanism for job shop schedules". In: *The International Journal of Advanced Manufacturing Technology* 22.9-10, pp. 669–680 (cit. on p. 20).

Subramaniam, V., A. Raheja, and K. Rama Bhupal Reddy (2005). "Reactive repair tool for job shop schedules". In: *International Journal of Production Research* 43.1, pp. 1–23 (cit. on p. 22).

Taillard, E. (1993). "Benchmarks for basic scheduling problems". In: *european journal of operational research* 64.2, pp. 278–285 (cit. on pp. 4, 30).

Tay, J. C. and N. B. Ho (2008). "Evolving dispatching rules using genetic programming for solving multi-objective flexible job-shop problems". In: *Computers & Industrial Engineering* 54.3, pp. 453–473 (cit. on p. 21).

Van de Vonder, S., E. Demeulemeester, and W. Herroelen (2008). "Proactive heuristic procedures for robust project scheduling: An experimental analysis". In: *European Journal of Operational Research* 189.3, pp. 723–733 (cit. on p. 20).

Vergnano, A., C. Thorstensson, B. Lennartson, P. Falkman, M. Pellicciari, F. Leali, and S. Biller (2012). "Modeling and Optimization of Energy Consumption in Cooperative Multi-Robot Systems". In: *IEEE Transactions on Automation Science and Engineering* 9.2, pp. 423–428. ISSN: 1545-5955. DOI: 10.1109/TASE.2011.2182509 (cit. on pp. 5, 35).

Vieira, G. E., J. W. Herrmann, and E. Lin (2000). "Analytical models to predict the performance of a single-machine system under periodic and event-driven rescheduling strategies". In: *International journal of production research* 38.8, pp. 1899–1915 (cit. on p. 20).

Vieira, G. E., J. W. Herrmann, and E. Lin (2003). "Rescheduling manufacturing systems: a framework of strategies, policies, and methods". In: *Journal of scheduling* 6.1, pp. 39–62 (cit. on pp. 5, 19).

Wigström, O. and B. Lennartson (2013). "Integrated OR/CP optimization for Discrete Event Systems with nonlinear cost". In: *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*. IEEE, pp. 7627–7633 (cit. on p. 28).

Wigström, O., B. Lennartson, A. Vergnano, and C. Breitholtz (2013). "High-level scheduling of energy optimal trajectories". In: *Automation Science and Engineering, IEEE Transactions on* 10.1, pp. 57–64 (cit. on p. 35).

Wigström, O., N. Murgovski, S. Riazi, and B. Lennartson (2017). "Computationally efficient energy optimization of multiple robots". In: *Automation Science and Engineering (CASE), 2017 IEEE International Conference on*. IEEE (cit. on p. 6).

Wilson, J. M. (2003). "Gantt charts: A centenary appreciation". In: *European Journal of Operational Research* 149.2, pp. 430–437 (cit. on p. 10).

Wu, S. D., E.-S. Byeon, and R. H. Storer (1999). "A graph-theoretic decomposition of the job shop scheduling problem to achieve scheduling robustness". In: *Operations Research* 47.1, pp. 113–124 (cit. on p. 20).

Wu, S. D., R. H. Storer, and P.-C. Chang (1992). "A rescheduling procedure for manufacturing systems under random disruptions". In: *New directions for operations research in manufacturing*. Springer, pp. 292–306 (cit. on p. 20).