One-Machine Scheduling with Unit Jobs and Chain Precedence Relations

Mara Servilio
One-Machine Scheduling with Unit Jobs and Chain Precedence Relations
# Contents

Acknowledgements iii  

1 Introduction 1  

1.1 A competitive scheduling problem in UMTS systems 5  
1.2 A competitive scheduling problem in Computational Biology 10  
1.3 Time-Indexed Formulation of Competitive Scheduling Problems 13  

2 One-machine Scheduling Problem in UMTS Channel Assignment 17  

2.1 The problem 17  

2.1.1 Problem formulation 17  
2.1.2 Problem complexity 19  

2.2 Solution algorithms 21  

2.2.1 A dynamic programming algorithm for $\beta$-FCFS 21  
2.2.2 A fast lagrangian heuristic for $\beta$-FCFS and M-FCFS 22  
2.2.3 An iterative greedy-like heuristic 28  

2.3 Computational experience 29  

3 Polyhedral study of one-machine scheduling problem: the case of a single user 33  

3.1 Introduction 33  
3.2 PSP$_1$ 38  
3.3 CSP$_1$ 46  

4 Polyhedral study of one-machine scheduling problem: the case of two users 49  

4.1 PSP$_2$ 49  
4.2 SP$_2$ 78  
4.3 CSP$_2$ 79
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Conclusions</td>
<td>87</td>
</tr>
</tbody>
</table>
Acknowledgements

There are many people which supported my activity in the last three years and I want to spend few words of thanks for them.
First of all, I wish to thank my tutor Claudio Arbib for passing me his enthusiasm for the research. I thank him for all precious comments and advice, for his encouragement and support, and for indispensable help in the realization of this work. I'm also grateful to Martine Labbé for giving me the opportunity to spend four pleasant months in Bruxelles. I learned a lot from her and a considerable part of this thesis is due to her contribution. I thank my advisor Dario Pacciarelli and Professor Fernando Nicolò for their help, especially in the first months of my doctoral course: they made for my complete integration with the research group of the University of Rome. My special thanks are due to Gaia Nicosia, Fabrizio Rossi and Stefano Suriglio for their friendship and helpfulness in answering to all my questions. They have been a very important guide for me!

There are several other people I'm grateful to: Monia, Fabrizio and Salvatore for useful suggestions during our discussions on various topics; all the PhD students of L’Aquila, for making pleasant these three years; Marta, Ludovica and Alessandra, for our continuous scientific exchanges and for their moral support before I gave my talks. They are great friends for me!

Finally, I want to thank my family for having total trust in all my choice, and Fabio for his infinite patience and love.

Rome, April 30, 2005

Mara Servilio
Chapter 1

Introduction

The scheduling problems arise whenever one has to allocate resources to activities over time. Originally, these problems were solved by general mathematical programming tools (linear programming, flows) but, in the seventies, the development of the computational complexity theory allowed to solve scheduling problems in a systematic way, by studying their complexity, some resolutive solution algorithms and their applications in to real life.

In general, when defining a scheduling problem one specifies the features of the system (i.e. the available resources), of the jobs (i.e. the activities which have to be performed) and of the objectives that one wants to optimize.

Most classical papers on scheduling relate to deterministic, off-line models, and are based on particular assumptions which are not always compatible with the real features of the problem; that explains why, in the last few years, it was begun to consider on-line and stochastic scheduling models taking into account the real aspects of the problem.

In a classical approach, the jobs belong to a single user which wants to organize and exploit the available resources in the most useful way: in particular there is a single objective function which has to be optimized. Also in the case in which there are more objectives, the resolutive solution algorithms are designed in a perspective in which a single decision-maker seeks for an optimal schedule of the system.

A very different situation is when there are more agents, each with its own set of jobs, in competition for the same resources and there is no “authority” able to solve possible conflicts among them. In this case, it is necessary to redefine, at least in part, the model so as to offer the possibility of a negotiation among the agents in such way that they can obtain a resource allocation admissible.

Many disciplines such as Combinatorial Optimization, Game Theory, Artificial Intelligence and Decision Theory provide an approach to multi-agent scheduling problems; in fact, the study of these models has been frequently
motivated by the real requirements in different fields. For example, in [3] the Authors consider the problem in which two different railway societies have to negotiate the use of a common track. Each society has a set of trains and a fixed timetable to respect: the objective of each agent consists in minimizing the sum of the delay for his own trains.

In multi-agent scheduling models, a fundamental difference arises depending on whether it is possible to transfer utilities among the agents or not; this condition typically means that all the users which are penalized from a certain allocation can be balanced by money. In this work, we consider the case in which the transfer is not allowed and analyze a particular class of Combinatorial Optimization problems known as Competitive Scheduling problems. In particular, we will discuss more in detail the situation when two users compete for the same single resource. Approaching such problem, we can adopt two slightly different viewpoints: we can optimize the performance of one user by according to the other an utility at least equal to a reasonable fixed threshold, or we can determine the set of all the nondominated schedules, i.e. such that a better schedule for one user necessarily corresponds to a worse schedule for the other. Since we consider a single shared resource, the problem is analogous to a single-machine competitive scheduling problem. Observe that for competitive scheduling problems we consider again a single decision-maker, unlike most of multi-agent scheduling problems.

We introduce the general notation for Competitive Scheduling. Let $N$ be the set of agents, each of them having a set of nonpreemptive jobs to process; for each agent $i \in N$, we denote $J_i$ its own set of jobs and $J^j_i$ its own $j$-th job whose processing time is indicated with $p^j_i$. Let $n_i = |J_i|$; depending on the situation, each job can be associated with other parameters as a due-date $d^i_j$ (i.e. the time by which the job $j$ should be completed), a release date $r^i_j$ (i.e. the time before which the job $j$ cannot start processing), a weight $w^i_j$ (i.e. the profit/cost for each time unit in which the job $j$ is in progress). A general schedule $\sigma$ is an assignment of starting times to each job and $C^j_i(\sigma)$ denotes the completion time of job $J^j_i$ in the schedule $\sigma$. Each agent $i$ is associated with an utility function $f^i(\sigma)$ which is non-decreasing when the completion times increase and the objective of each user consists in maximizing its own utility function. In certain cases, this function can be replaced with a cost function and then the objective becomes to minimize it. When necessary, we will use the classical three-field notation for the scheduling problems $\alpha|\beta|\gamma$, where $\alpha$ identifies the system and the number of machines and contains a single entry, $\beta$ provides the (possible) particular features of the system and may contain no entries, a single entry or multiple entries, and $\gamma$ identifies the objective function and usually contains a single entry.
Possible machine environments specified in the α field are

- Single machine (α = 1);
- Identical machines in parallel (α = P_m);
- Machines in parallel with different speeds (α = Q_m);
- Flow shop (α = F_m): there are m machines in series and each job has to be processed on each one of the m machines. All the jobs have the same routing for the processing;
- Job shop (α = J_m): there is the same situation as the flow shop, but each job has its own route to follow for the processing.

Possible entries in the β field are

- Release dates (β = r_j);
- Preemptions (β = prmp): preemptions imply that it is not necessary to keep a job on a machine until completion;
- Precedence constraints (β = prec);
- Breakdowns (β = brkdwn): breakdowns imply that machines are not continuously available.

Finally, possible objective functions specified in the γ field are

- Makespan (γ = C_{max}): the makespan is equivalent to the completion time of the last job to leave the system;
- Total weighted completion time (γ = \sum_j w_j C_j);
- Total weighted tardiness (γ = \sum_j w_j T_j): the tardiness is defined as T_j = \max\{C_j - d_j, 0\}, where C_j is the completion time of job j and d_j is its due-date;
- Weighted number of tardy jobs (γ = \sum_j w_j U_j).

In the literature, very important contributions regarding Competitive Scheduling have been recently provided by [1] and [2]. In [1], the Authors consider the scenario where two users compete to perform their respective jobs on a common set of resources: they describe the set of nondominated schedules for which the users may negotiate and develop a polynomial algorithm to find this nondominated set. In [2], the same Authors consider the problem of scheduling general length jobs with no-precedence relations on a single machine, with
the objective of best compromising between two regular functions of the job completion times. They characterize the complexity of decision problems arising for different combinations of the objective functions for the two users and different system structures and, for some special cases, they enumerate all the nondominated solutions.

A similar argument has been approached in [4], where the Authors analyze a one-machine scheduling problem in which two or more agents are involved and assume that, unlike most scheduling problems, each agent has its own objective. This is not the case of traditional models, in which different jobs represent different users but individual requirements of each user do not affect the scheduling criterion. Instead, in [4] the model is characterized by multiple scheduling criteria and different objectives each of them associated with a class of agents. In other words, each agent self-selects according to its own preferred criterion and the objective consists in optimizing a weighted average of the criteria. In this work, three main criteria are considered: minimizing makespan, minimizing maximum lateness and minimizing total weighted completion time and it is proved that, unlike the case in which only one of these criteria is considered, when a mix of them is minimized the problem becomes NP-hard. Further, some dominance properties which hold for the problem have been provided.

On the other hand, we mention that multi-agent scheduling problems have been extensively studied by [13] and [16] in the manufacturing systems; in this area, all the elements of the manufacturing process (machines, jobs, workers...) can play the role of agents, each of them having the aim to maximize its own productivity.

In [20] a decentralized scheduling problem is considered; it consists in allocating resources for distributed computing systems whose nature of computation can be decentralized. For example, think to a problem of scheduling network access for programs representing different users on the internet scenario: all the modules of the system represent independent users, each of them having competing requirements and localized informations about their needs. The independence among the users can be managed by treating the modules as agents which have the autonomy to decide how exploit their resources and which can communicate each other, by messages, some of their private informations. A decentralized solution is to manage message passing, to reach closure and to realize the final schedule. These problems can be solved by market mechanisms; in [20], the Authors show how some results in economic theory can be applied to decentralized scheduling, describe a specific problem of this kind and provide a formal economic model of it.

This research was motivated by two main applications. The first concerns a two-users competitive scheduling problem arose in a Uni-
UCMersal Mobile Telecommunication System (UMTS) developed within the European IST project FUTURE [6]. The project contributes to the specication of the 3rd generation mobile communication system developed by the European Telecommunications Standard Institute. In particular, it aims at adopting the recent advances of the internet scene in UMTS by exploring the applicability of native internet protocols in 3G, and investigating and developing enhanced Quality of Service (QoS) strategies for a packet based UMTS.

We consider the procedures in charge of assigning the channel capacity to the mobile terminals (users). These procedures are part of the scheduling function implemented in downlink by the system. Since the system includes two users, a competitive scheduling problem arises in which one wants to maximize or the on-time packets transmitted to one user while guaranteeing certain amounts of on-time packets to the other, or the total on-time traffic for each user, or the worst between the total on-time traffic of the two users.

The second application concerns an alignment problem in Computational Biology, where the main objective is to compare two (or more) biological objects which consist of a set of elements arranged in a linearly ordered structure and to align them by determining subsets of corresponding elements in each. Typical examples are the alignments of genomic sequences or proteins.

1.1 A competitive scheduling problem in UMTS systems

The introduction of UMTS telecommunication systems is a very innovative event which allows to reach a lot of objectives such that the convergence between fixed and mobile networks and the availability of a large range of services called “multimedia communication”. The main functionality of UMTS networks is the capacity to support wideband service data, symmetric and asymmetric communications, real-time and non-real-time services and packet switched traffic which allows to simultaneously transmit data with different bit-rates.

Since a UMTS system satisfies a lot of services having very different features, it is essential that the network associates each of them with a certain Quality of Service which allows to univocally identify the requirements of the transport service that one can use.

In a UMTS system it is possible to distinguish four different QoS classes:

- **Conversational Class**: the main services provided by this class are the communications among two or more users. Then, it is used to transmit real-time data and is very susceptible to the transfer time; in particular, for this class a very short transfer delay is tolerated since the quality requirements are determined by the human perceptions.
• Streaming Class: it is used to transmit real-time data both video and audio kinds and it allows a transfer delay greater than the conversational class.

• Interactive Class: the typical services satisfied by this class are the web browsing, the research in a database and the access to a certain server. Then, the main requirements of the interactive class are a reasonable transfer delay and a transparent data transmission having a minimum error rate.

• Background Class: it is used to satisfy services such as e-mail and SMS transfer or files download. Since they do not belong to a real-time traffic, this class is not as susceptible to the transfer delay as the others, but it requires a maximum reliability and integrality of the data transmission.

A UMTS system operates via protocols associated with distinct layers, so that the communication within a layer is transparent with respect to all the lower level layers. The UMTS radio interface is made up of three main layers:

• Physical Layer (L1)

• Data Link Layer (L2)

• Network Layer (L3)

In particular, in the layer L2 we distinguish further sub-levels as Medium Access Control (MAC), Radio Link Control (RLC), Packet Data Convergence Protocol (PDCP) and Broadcast/Multicast Control (BMC).

The physical layer (PHY) is the lowest layer in the ISO/OSI reference model and it supports all the functionalities allowing the transmission on the physical stratum; in particular, it provides the transfer of the useful traffic to MAC and all the higher levels.

In this work, we consider the communication from MAC module to physical layer. We distinguish between the downlink and the uplink communication depending on whether a message is transmitted from MAC to physical layer at gateway side or at user side, respectively.

The data transfer between MAC and PHY protocols occurs via transport channels which are associated with a transport format, i.e. an appropriate combination of scrambling, interleaving and bit rate to apply to each information which has to be transmitted.

We focus on the communication from gateway to user (in the following, the downlink communication) at the gateway side (white arrow in Figure 1.1a); the data are transmitted by a unique transport channel, named Downlink.
1.1. A COMPETITIVE SCHEDULING PROBLEM IN UMTS SYSTEMS

Shared CHannel (DSCH). In the FUTURE demonstrator, the message format relevant to this communication is the one reported in Figure 1.1b.

![Diagram](image)

Figure 1.1: a) layers involved and b) message format in the downlink communication between MAC and PHY.

The useful traffic information is contained in the DSCH section of the message. According to the traffic load, the length of this section can be 5120, 2560 or 0 bits. Every Transmission Time Interval (TTI, corresponding to 40 msec) a
message exchange occurs between the MAC and the PHY protocols, see Figure 1.1a, yielding a bit rate of 128, 64 or 0 Kbit/sec depending on the different length of the DSCH sections. For further details, see [5].

A relevant issue in FUTURE is that, unlike previous projects on the same subject, different services and applications can simultaneously be provided to two mobile users (say A and B) by a unique physical channel per satellite beam. Since the services may have different QoS requirements, a need arises to coordinate the radio resource utilization. Such a need is particularly demanding in downlink because in this case the DSCH can be accessed by both mobile terminals, and not only has the scheduler to select suitable transport formats to maximize the ratio between useful traffic and signalling, but also has to specify how the channel is to be shared. Since multiplexing is performed using orthogonal variable spreading factor codes, the DSCH capacity can be assigned in four ways only: either the channel is completely dedicated to one user, or completely dedicated to the other, or equally divided between the two, or finally not assigned. Note that the access mode to the DSCH (at most two users per TTI) can be applied to an arbitrary number of active users: far from being a limit, this practice avoids an excessive packet segmentation and the derived overhead.

Deciding at each TTI how to assign the DSCH capacity to users is the task of a module called Capacity Planner. Figure 1.2 illustrates the interactions between the Capacity Planner and the other modules of the FUTURE scheduling function, in particular the Short-term Scheduler which is in charge of packet segmentation and of selecting the most suitable transport format for the communication.
Figure 1.2: System architecture: interactions between the Capacity Planner and other modules.
Packets are stored in two groups of 13 queues (logical channel) each. A group corresponds to a user and a logical channel to a specific application type. For each user, the Capacity Planner constructs blocks of 2560 bits (corresponding to half channel capacity, 64Kbit/sec) by selecting packets from the logical channels; then, depending on the traffic, it assigns a number of blocks between 0 and 2 to each TTI. Each packet is associated with its time-to-leave, that is, with the number of TTIs that may expire before the packet is considered late, and the arrival time of the packet plus the time-to-leave defines the packet due-date. Once a block is assigned to a TTI, the number of on-time packets (or the total number of on-time bits) is completely determined. Since this measure can be associated with every pair (block, TTI), computing an optimum plan means finding an assignment of blocks to TTIs which maximizes the on-time traffic of the two users. This step involves the solution of a competitive scheduling problem which is the subject of our study.

Whatever be the algorithm employed by the Capacity Planner to schedule the channel to the users, it must agree to the following specific implementation requirement:

The order in which data-packets are sent by the transmitter (gateway) must be preserved at the receiver (mobile terminal), because no two packets relevant to the same application can be swapped in time.

Since blocks are formed by taking the packets of each application from the relevant queue in their order of arrival, a strict order is defined among the blocks of each user (the blocks of the two users are however ranked independently on each other). Then, by the above requirement any feasible schedule of blocks to TTIs must fulfill two conditions:

No-cross Condition. The input and output orders are to be consistent: if block \( i \) is assigned to the \( s \)-th TTI of the planning horizon, block \( j \) to the \( t \)-th, and \( i \) precedes \( j \), then necessarily \( s < t \).

No-loss Condition. No block is to be lost: if block \( i \) is assigned to the \( s \)-th TTI of the planning horizon, block \( j \) to the \( t \)-th, and there is a block \( k \) between \( i \) and \( j \), then \( k \) must necessarily be assigned to some TTI (between the \( s \)-th and the \( t \)-th).

In this thesis, we will study the combinatorial aspects of the one-machine competitive scheduling problem illustrated in this section.

1.2 A competitive scheduling problem in Computational Biology

Computational Biology expanded greatly in the last years, following the con-
1.2. A COMPETITIVE SCHEDULING PROBLEM IN COMPUTATIONAL BIOLOGY

siderable development of computers which are adopted in several biology laboratories to store and manage genomic data; in fact, the use of computers allows to very quickly complete a lot of projects in this field that otherwise would require long time. One of the most important projects in Computational Biology was the *Human Genome Project* which ended in 2001 in the announcement of the sequencing of the human genome.

Actually, we distinguish between bioinformatics problems which concern storage, organization and distribution of large amounts of genomic data, and Computational Biology which is implemented to solve problems of interpretation and analysis of genomic data. In particular, this young science allows to model biological processes in the cell, to remove experimental errors from genomic data, to interpret data and to provide theories regarding their biological interactions.

After that biological problem is reformulated in mathematical terms, Combinatorial Optimization and, particularly, Operations Research is adopted to solve it, especially in the case in which this problem is NP-hard. Branch-and-Cut, Branch-and-Price and Lagrangian Relaxation are the most adopted techniques for Computational Biology problems formulated by Integer Linear or Quadratic Programming.

In this thesis, we study a problem whose a particular case, well-known as Alignment Problem, has been extensively examined in Computational Biology. One can distinguish between sequence and structure alignment problem: the former consists in comparing more biological objects comprised of elements which are arranged in a linearly ordered structure, and in finding a correspondence among the elements in each object. This correspondence must preserve the order of elements: this means that, if \( i \)-th element in the first object is in correspondence with \( j \)-th element in the second object, then a correspondence between an element following \( i \) and an element preceding \( j \) cannot exist.

In mathematical terms, given two biological objects, the former comprised of \( n \) elements and the latter of \( m \), we consider the complete bipartite graph \( G_{nm} = (V, U, E) \), where \(|V| = n\), \(|U| = m\) and \( E = V \times U \). An alignment corresponds to a noncrossing matching on \( G_{nm} \), i.e. a matching in which no two edges cross.

On the other hand, a typical structure alignment problem in Computational Biology is the Contact Map Overlap (COM) problem, which aims to compare the structure of different proteins. The proteins are linear chains of amino acids, or residues, whose type and sequence guarantee a natural distinction among them. Each protein folds into a three-dimensional structure and, in this way, it determines its functioning and interaction with other molecules. The comparison among proteins belonging to very different families allows to identify functional and structural similarity which coulds contain a lot of in-
formations about their common evolution. In order to compare two different structures, it is necessary to align them in some way. We observe that sequence and structure alignments are computationally very different: in fact, the structure alignment is a three-dimensional computational problem and is more complex than the sequence alignment which is one-dimensional. In general, when a protein folds it may happen that two residues which were not adjacents in the original sequence, are very close each other in the new space. The three-dimensional structure of a general protein with $n$ residues is represented by a contact map, i.e. a 0-1, symmetric, $n \times n$ matrix whose entries equal 1 in correspondence with a pairs of residues which are in contact, i.e. that are within a fixed small distance in the protein’s fold, but are not adjacents in the original sequence. Then, given two folded proteins, the problem consists in determining an alignment between their residues in order to establish which among them are equivalents. The aim is to determine the best alignment in correspondence of which there is the largest set of common contacts.

In terms of graphs, we consider two undirected graphs $G_1 = (V, E_1)$ and $G_2 = (U, E_2)$ and denote a general edge as an ordered pair $(i, j)$. An alignment of $V$ and $U$ corresponds to a noncrossing matching $M$ on $G_{nm}$. If two edges $e = (v_1, v_2) \in E_1$ and $f = (u_1, u_2) \in E_2$ are such that lines $l = \{v_1, u_1\}$ and $h = \{v_2, u_2\}$ belong to $M$, then we say that $l$ and $h$ generate the sharing $(e, f)$. The CMO problem consists in finding an alignment which maximizes the number of sharings. Exploiting the noncrossing condition for the alignment problem, it is possible to define the compatibility among sharings. Given edges $e = (v_1, v_2)$, $e' = (v'_1, v'_2) \in E_1$ and $f = (u_1, u_2)$, $f' = (u'_1, u'_2) \in E_2$, we say that sharings $(e, f)$ and $(e', f')$ are compatible if lines $\{v_1, u_1\}$, $\{v_2, u_2\}$, $\{v'_1, u'_1\}$ and $\{v'_2, u'_2\}$ are noncrossing. At this point, if one defines a new graph $G_{SC}$ (Sharings Conflict Graph) where each node is associated with a pairs $(e, f)$, for each $e \in E_1$ and $f \in E_2$, and there exists an edge between two nodes if and only if the corresponding pairs are not compatible sharings, then the CMO problem can be reduced to a Stable Set problem on $G_{SC}$. In [11], the Authors provide a rigorous algorithm for structure comparison; in particular, they develop an effective integer linear programming formulation for CMO problem and a Branch-and-Cut strategy to solve it; the main innovation of this work is the effective use of integer programming method in the field of biomolecular structure comparison.
1.3 Time-Indexed Formulation of Competitive Scheduling Problems

A lot of scheduling problems are formulated as integer programs by using time-indexed variables, i.e. variables indexed by \( \{i, t\} \) where \( i \) denotes a job and \( t \) a time period in which the job can be processed. In this work, we investigate the time-indexed formulation for a nonpreemptive single-machine scheduling problem with precedence constraints and emphasize the competitive aspect.

A time-indexed formulation for nonpreemptive single-machine scheduling problems has been widely studied in [17], where the Authors examine the general problem of choosing jobs over time to allocate on a single machine, taking into account the resource constraints. The time-indexed formulation is

\[
\begin{align*}
\max / \min \sum_j \sum_t f_j(t)x_{jt} \\
\sum_t x_{jt} & \leq 1 \quad \forall j \\
\sum_j \sum_s a_j(t-s)x_{js} & \leq b_t \quad \forall t \\
x_{jt} & \in \{0, 1\} \quad \forall j, \forall t
\end{align*}
\]

where \( x_{jt} \) equal 1 if job \( j \) is started in period \( t \) and 0 otherwise, \( f_j(t) \) denotes the profit/cost associated with the allocation of job \( j \) in time \( t \), \( a_j(t-s) \) is the amount of resource used by job \( j \) after \( t - s \) periods and \( b_t \) is the available amount of resource in period \( t \).

A special version of this problem is studied in [18], where \( b_t \) equal 1 for each period \( t \), \( p_j \) denotes the processing time of general job \( j \) and \( a_j(t-s) \) equal 1 if \( 0 \leq t - s < p_j \) and 0 otherwise. Given a certain time horizon \( T \), the scheduling model is
Further special cases can be considered if due-date $d_j$ and release date $r_j$ are introduced for each job $j$; also different objectives, such as weighted start times or weighted tardiness, can be examined.

The first constraint in the above model means that all the jobs in the system must be assigned to exactly one period in the time horizon; often, this constraint makes difficult the study of the problem, especially in a polyhedral approach. For this reason, in [18] the Authors relax these inequalities by substituting them with

$$\sum_{t=1}^{T-p_j+1} x_{jt} \leq 1 \quad \forall t \leq T$$

The new constraint means that each job can be assigned to at most one period. Observe that the relaxed time-indexed formulation of this scheduling problem is characterized by set packing constraints.

Frequently, in the real time applications it is necessary to assume that the jobs must fulfill some precedence relations among them; in [15], the Authors express the fact that job $j$ precedes job $k$ in each feasible schedule by introducing either the condition

$$x_{jt} - \sum_{s=t+p_j}^{T-p_k+1} x_{ks} \geq 0 \quad \forall t = 1, \ldots, T - p_j + 1$$

or the inequality

$$\sum_{t=1}^{T-p_j+1} (t-1)x_{jt} + p_j - \sum_{t=1}^{T-p_k+1} (t-1)x_{kt} \leq 0$$

Plugging one of them into the original problem, the resulting model is characterized by set packing inequalities plus an add constraint.
In this work, we analyze two different competitive scenarios and provide a time-indexed formulation for both of them.

We consider \( n \) users competing to assign their own jobs \( J^k = \{1, \ldots, n_k\} \), \( k = 1, \ldots, n \), over time on a common single machine. In the first scenario, we suppose that \( n = 2 \) and maximize \( f^1(\sigma) \) by guaranteeing that the value \( f^2(\sigma) \geq b \), where \( b \) is a fixed non-negative threshold (if we minimize the objective function, then \( f^2(\sigma) \leq b \)). For each job \( j \in J^k \), \( k = 1, 2 \), we define a processing time \( p^k_j \) and a profit/cost \( f^k_j(\tau) \) associated with the assignment of job \( j \in J^k \) to period \( t \). If we assume that not all the jobs must be assigned (i.e. an optimal schedule may not be complete), then the time-indexed formulation of the problem is

\[
\begin{align*}
\max & \sum_{j \in J^1} \sum_{t = 1}^{T - p^1_j + 1} f^1_j(t)x_{jt} \\
& \sum_{j \in J^2} \sum_{t = 1}^{T - p^2_j + 1} f^2_j(t)y_{jt} \geq b \\
& \sum_{t = 1}^{T - p^1_j + 1} x_{jt} \leq 1 \quad \forall j \in J^1 \\
& \sum_{t = 1}^{T - p^2_j + 1} y_{jt} \leq 1 \quad \forall j \in J^2 \\
& \sum_{j \in J^1} \sum_{s = t - p^1_j + 1}^{t} x_{js} \leq 1 \quad \forall t = 1, \ldots, T \\
& \sum_{j \in J^2} \sum_{s = t - p^2_j + 1}^{t} y_{js} \leq 1 \quad \forall t = 1, \ldots, T \\
\end{align*}
\]

where \( x_{jt} \) and \( y_{jt} \) are the variables associated with user 1 and 2, respectively.

Also this model is characterized by set packing inequalities plus an add constraint (the first, in the above formulation) which is, in this case, a knapsack constraint.

In the second scenario, we solve the scheduling problem by maximizing/minimizing the amount \( f^1(\sigma) + f^2(\sigma) \). Then, the knapsack constraint disappears from the time-indexed formulation, the objective function is modified and the model is described by set packing constraints, only.
Then, the correspondence between time-indexed one-machine scheduling (1S) and set packing (SP) formulations is as follows

<table>
<thead>
<tr>
<th>1S</th>
<th>⇔</th>
<th>SP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1S + prec</td>
<td>⇔</td>
<td>SP + add constraint</td>
</tr>
<tr>
<td>1S + prec + competitive relations</td>
<td>⇔</td>
<td>SP + knapsack constraint</td>
</tr>
</tbody>
</table>

In this thesis, unlike in [18] and [15] we will study both the competitive versions of the one-machine scheduling problem above introduced. Moreover, we will assume that each job has a unit processing time and include chain-like precedence constraints; in [15], the Authors studied the non-competitive version with general processing times and general precedence relations, but for the special model whose formulation is in terms of completion times.

The work is divided into two fundamental parts. In the former (Chapter 2), we will study the complexity of the first scenario of the competitive scheduling problem and present different resolutive approaches to solve it and its particular cases, also taking into account the real-time constraints originated by the applications. In the latter (Chapters 3, 4), we will propose a polyhedral study of the second scenario of the problem. In particular, after that different cases have been listed, we examine the dimension of the polyhedron defined as the convex hull of feasible assignments of each problem and determine some facet-defining inequalities for it.
Chapter 2

One-machine Scheduling Problem in UMTS Channel Assignment

In this chapter we formulate and discuss the competitive scheduling problem solved by the Capacity Planner in the UMTS application, as described in §1.1.

2.1 The problem

In §2.1.1 two general formulations of the problem are specialized through properties of objectives and solutions inherited by the application. In §2.1.2 the complexity of the resulting problems is investigated.

2.1.1 Problem formulation

Let \( A, B \) be disjoint sets of \( n_1, n_2 \) jobs with unit processing times and \( T \) be a set of \( m \) unit time slots. In our application,

- each set of jobs corresponds to a distinct user (\( A \) or \( B \));
- a job corresponds to a block of segmented data-packets: packets in a block may belong to different applications activated by the relevant user, and may have different due dates;
- each time slot corresponds to half TTI (20 ms).

From now on we will assume that a strict order is defined between the elements of \( A \) (of \( B \), of \( T \)).

The function \( \sigma : A \cup B \to T \) represents a competitive schedule which assigns a set of jobs to a set of slots. In a feasible schedule not all the jobs (the
slots) have to be necessarily assigned, that is, a schedule may not be complete. Schedules are not equally preferable: whenever job \( j \in A \) (respectively, \( j \in B \)) is assigned to slot \( t \in T \), a non-negative profit \( a_j(t) \) (respectively, \( b_j(t) \)) is gained by the corresponding user. As explained in §1.1, such a profit equals the amount of on-time data-packets (or of on-time bits of useful traffic) sent when block \( j \) is assigned to the \( t \)-th time slot of the planning horizon. Since a larger and larger amount of information in an unscheduled block is delayed as time passes, to model packet delay it is sufficient to focus on profits non-increasing with \( t \in T \). Thus we will make the following first general assumption:

**Assumption 2.1.1.** For any job \( j \in A \) or \( j \in B \), \( a_j(s) \geq a_j(t) \) and \( b_j(s) \geq b_j(t) \) for all \( s \) and \( t \) such that \( s < t \); that is, the functions \( a_j, b_j \) are regular non-increasing for all \( j \in A \cup B \).

To separately measure the quality of service of the two users, we qualify every schedule \( \sigma \) by two weights

\[
\begin{align*}
a(\sigma) &= \sum_{j \in A} a_j(\sigma(j)) \\
b(\sigma) &= \sum_{j \in B} b_j(\sigma(j))
\end{align*}
\]

Let in general \( S \) denote the set of feasible schedules. We focus on two main problems, according to whether one wishes (i) to optimize the worst QoS of the two users, or (ii) to provide one user at least with a prescribed QoS \( \beta \in \mathbb{R}_+ \).

**Problem 2.1.1.**

\[
\max_{\sigma \in S} \min \{ a(\sigma), b(\sigma) \} \tag{2.1}
\]

**Problem 2.1.2.**

\[
\max_{\sigma \in S} \{ a(\sigma) \mid b(\sigma) \geq \beta \} \tag{2.2}
\]

We can consider a third problem in which one wishes (iii) to optimize the sum of the total QoS of the two users.

**Problem 2.1.3.**

\[
\max_{\sigma \in S} \{ a(\sigma) + b(\sigma) \} \tag{2.3}
\]

However, since we will approach the polyhedral aspects of this last model, we refer to Chapters 3 and 4 for details about it.

Let us give the following definitions.
2.1. THE PROBLEM

**Definition 2.1.1.** A schedule $\sigma$ is active if for all $r, s, t \in T$ with $r < s < t$, $\sigma(i) = r$ and $\sigma(j) = t$ for some $i, j \in A \cup B$ imply $\sigma(k) = s$ for some $k \in A \cup B$.

**Definition 2.1.2.** A schedule $\sigma$ is first-come first-served with respect to an ordered set of jobs $J \subseteq A \cup B$ if

(i) for all $i, j \in J$, $\sigma(i) = r, \sigma(j) = t$ and $i < j$ imply $r < t$;

(ii) for all $i, j, k \in J$, $\sigma(i) = r, \sigma(j) = t$ and $i < k < j$ imply $\sigma(k) = s$ for some $s \in T$.

Independently on the problem considered, by Assumption 2.1.1 we can limit our attention to active schedules. Moreover, by the No-cross and No-loss Conditions of §1.1 every feasible schedule has to be first-come first-served with respect to both $A$ and $B$. Summarizing, we can make a second general assumption:

**Assumption 2.1.2.** The set $S$ of feasible solutions consists of all the first-come first-served active schedules of sets $J \subseteq A \cup B$. Since $a_j(t), b_j(t) \geq 0$ for all $j$ and $t$, we limit our attention to sets $J$ with $\bar{m} = \min\{n_1 + n_2, m\}$ elements.

The corresponding restricted Problem 2.1.1 and Problem 2.1.2 will be in the following referred to as:

- **Maximin First-Come First-Served Problem (M-FCFS);**
- **$\beta$ First-Come First-Served Problem ($\beta$-FCFS).**

2.1.2 Problem complexity

Problem 2.1.2 is NP-hard for general processing times and non-FCFS schedules [2]. However, in our case one may expect to exploit unit processing times and the properties of feasible schedules, as well as the regularity of $a_j(t), b_j(t)$, in the design of a polynomial exact algorithm. Unfortunately, these properties do not help simplify the problem:

**Theorem 2.1.1.** Problems 2.1.1 and 2.1.2 are NP-hard even when $a_j(t) = b_j(t) = w_t$ for all $j \in A \cup B$ and $t \in T$.

**Proof.** Let us first prove the theorem for Problem 2.1.2, by reduction from Subset Sum [7]: given a set $K$ of $n$ items with sizes $w_k, k = 1, \ldots, n$, and an integer $z$ find a subset $X$ of $K$ with $w \leq \sum_{k \in X} w_k \leq z$, where $w = \frac{1}{2} \sum_{k \in K} w_k$. Construct an instance of Problem 2.1.2 by setting $A = B = T = K$, $\beta = w$, and $a_j(t) = b_j(t) = w_t$ for any $t \in T$ and $j \in A$ or $j \in B$. Call $\sigma_X$ a solution
of this instance where the set $J$ of scheduled jobs of $B$ is assigned to $X \subseteq T$: then

$$b(\sigma_X) = \sum_{j \in B} b_j(\sigma_X(j)) \geq \beta$$

if and only if $\sum_{t \in X} w_t \geq w$. On the other hand,

$$a(\sigma_X) = \sum_{j \in A} a_j(\sigma_X(j)) = \sum_{t \in T - X} w_t = 2w - \sum_{t \in X} w_t \geq 2w - z$$

if and only if $\sum_{t \in X} w_t \leq z$. The reduction for Problem 2.1.1 is immediately derived from the formulation of Subset Sum as a maximization problem

$$\max_{X \subseteq K} \left\{ \sum_{k \notin X} w_k | \sum_{k \notin X} w_k \leq \sum_{k \in X} w_k \right\}$$

Proposition 2.1.1. For any schedule $\sigma_X$ of the instance of Problem 2.1.2 constructed in Theorem 2.1.1 there exists a complete FCFS schedule $\tilde{\sigma}_X$ assigning $J \subseteq B$ to $X$ and such that

$a(\sigma_X) = a(\tilde{\sigma}_X)$

$b(\sigma_X) = b(\tilde{\sigma}_X)$

In fact in the instance used in the reduction, the $b_j(t)$'s are all equal to $w_t$ ($t = 1, \ldots, n$). Thus we have the following

Corollary 2.1.1. Both problems M-FCFS and $\beta$-FCFS are NP-hard, even when $a_j(t) = b_j(t) = w_t$ for all $j \in A \cup B$ and $t \in T$.

Observation 2.1.1. Since we do not lose generality by assuming $w_1 \geq w_2 \geq \ldots \geq w_n$, both functions $a_j(t)$ and $b_j(t)$ used in the reduction are regular. Hence all the problems considered are NP-hard even if $a_j(t)$ and $b_j(t)$ respect Assumption 2.1.1 for all $j \in A \cup B$.

Observation 2.1.2. In the reduction one has $a_i(t) \leq a_j(t)$ and $b_i(t) \leq b_j(t)$ for any $t$ whenever $i < j$. Considering Assumption 2.1.1, it turns out that matrices $A = \{a_j(t)\}$ and $B = \{b_j(t)\}$ are double graded. Therefore, this property does not help simplifying the problem. We will however show (Proposition 2.2.3) that it can be exploited under additional assumptions.
2.2 Solution algorithms

In this section we describe (§2.2.1) a pseudo-polynomial dynamic programming algorithm for M-FCFS and $\beta$-FCFS, indicate special polynomial cases of Problem 2.1.2, illustrate (§2.2.2) fast lagrangian heuristics for M-FCFS and $\beta$-FCFS, and develop (§2.2.3) a very fast greedy-like heuristic for M-FCFS and $\beta$-FCFS to be used as a computational benchmark (see Section 2.3).

About the lagrangian heuristics, our approach is similar to that proposed by [9] for the constrained minimum spanning tree problem: based on particular properties of data and solutions, we develop an efficient combinatorial algorithm to solve the lagrangian dual and crossover to feasible solutions; but unlike [9], in our case the lagrangian approach does not lead to a guaranteed approximation, since one cannot exploit a matroid structure of the relaxed problem.

2.2.1 A dynamic programming algorithm for $\beta$-FCFS

Definition 2.1.2 has the following consequence on the optimal solutions of $\beta$-FCFS.

**Proposition 2.2.1.** Suppose that in an optimal solution of $\beta$-FCFS the scheduled jobs of $B$ are assigned to $S \subseteq T$, and let $S = \{s_1 < s_2 < \ldots\}$, $T - S = \{t_1 < t_2 < \ldots\}$. Then there exists an optimal solution which assigns the first job of $A$ (of $B$) to $t_1$ (to $s_1$), the second to $t_2$ (to $s_2$), and so on.

**Proof.** Since $\beta$-FCFS seeks for FCFS schedules, by (ii) of Definition 2.1.2 these will assign the first $|T - S|$ jobs of $A$. Let then $\sigma$ be an optimal solution, and let $j$ be the first job of $A$ assigned to $t_k$, $k \neq j$. As $\sigma$ is FCFS, one has $k > j$ and therefore $\sigma$ does not assign $t_j$ to any job in $A \cup B$. But since $a_j(t)$ is regular, $a_j(t_j) \geq a_j(t_k)$; hence a schedule $\sigma^*$ obtained from $\sigma$ by assigning job $j$ to $t_j$ is optimal as well. The same argument applies to $B$. \qed

Based on Proposition 2.2.1 we can solve $\beta$-FCFS by the following $O(\bar{m}^2 \beta)$ recursion, where $\bar{m} = \min\{n_1 + n_2, m\}$. Let $a(k, h, \gamma)$ denote the maximum value of a solution assigning $k$ jobs, $h$ of which belonging to $B$, and such that the profit for $B$ is $\geq \gamma$. Then

$$a(k, h, \gamma) = \max\{a(k - 1, h - 1, \gamma - b_h(k)), a(k - 1, h, \gamma) + a_{k-h}(k)\} \quad (2.4)$$

with the stipulation that $a(k, h, \gamma)$ is not defined if such a solution does not exist, and under the initial conditions

$$a(k, k, \gamma) = 0 \quad (2.5)$$
for all \( k \) and \( \gamma \) such that \( \gamma \leq \sum_{j=1}^{k} b_j(j) \) (a necessary condition for the existence of at least one solution returning to \( \mathcal{B} \) a profit \( \geq \gamma \)).

M-FCFS can be solved through formulæ (2.4)-(2.5) by halting the recursion as soon as \( a(k, h, \gamma) \geq \gamma \). The recursion can be extended to \( u > 2 \) users, at the price of increasing the complexity to \( O(\bar{m}^u \beta^{u-1}) \).

Formulae (2.4)-(2.5) give an efficient exact algorithm under specific assumptions:

**Proposition 2.2.2.** M-FCFS and \( \beta \)-FCFS can be solved in \( O(\bar{m}^2n_B) \) time when, for any \( j \in \mathcal{B} \), \( b_{jt} = b \geq 0 \) for \( t \in T \).

**Observation 2.2.1.** Proposition 2.2.2 can easily be generalized to the case \( b_{jt} = b \geq 0 \) for \( t \leq d_j \) and \( b_j(t) = 0 \) for \( t > d_j \).

Moreover, we have the following

**Proposition 2.2.3.** Under the assumption of Proposition 2.2.2, an optimal schedule fulfilling (i) but not necessarily (ii) of Definition 2.1.2 can be found in \( O(\bar{m}^2n_B) \) time when \( \mathbf{A} = \{a_j(t)\} \) is double graded.

**Proof.** It is easy to see that if \( \mathbf{A} \) is double graded, then there always exists an optimal solution which fulfils (ii) of Definition 2.1.2, and can therefore be found by formulæ (2.4)-(2.5). \( \square \)

### 2.2.2 A fast lagrangian heuristic for \( \beta \)-FCFS and M-FCFS

We next present a lagrangian heuristic for \( \beta \)-FCFS and M-FCFS. For the general description of lagrangian heuristics we refer to [8]. This approach looks suitable whenever the (NP-hard) problem consists of a polynomially solvable problem with some side constraints. Although \( \beta \)-FCFS and M-FCFS fall within this category, the real-time requirement can represent a severe barrier for the applicability of lagrangian heuristics. In this section we illustrate how the properties of \( \beta \)-FCFS and M-FCFS can be exploited to speed-up the key steps of the method so as to cope with the real-time requirement.

Let \( \mathcal{S} \) be the set of all complete FCFS schedules of \( \mathbf{A} \cup \mathbf{B} \). The two problems are:

\[
\max\{a(\sigma) \mid \sigma \in \mathcal{S}, b(\sigma) \geq a(\sigma)\} \quad \text{(M-FCFS)} \\
\max\{a(\sigma) \mid \sigma \in \mathcal{S}, b(\sigma) \geq \beta\} \quad \text{(\( \beta \)-FCFS)}
\]

The lagrangian relaxations of \( \beta \)-FCFS and M-FCFS obtained dualizing the side constraint read as follows:

\[
L_1(\lambda) = \max\{(1 - \lambda)a(\sigma) + \lambda b(\sigma) \mid \sigma \in \mathcal{S}\} \quad (2.6)
\]
2.2. SOLUTION ALGORITHMS

\[ L_2(\lambda) = \max \{ a(\sigma) + \lambda b(\sigma) - \lambda \beta \mid \sigma \in S \} \]  \hspace{1cm} (2.7)

and \( \min_{\lambda \geq 0} L_i(\lambda) \) is the lagrangian dual problem.

We show how the following three tasks can be performed efficiently:

1. solving the lagrangian relaxations (2.6)-(2.7) for any given \( \lambda \geq 0 \);
2. adjusting one solution to (2.6)-(2.7), for any fixed \( \lambda \geq 0 \), so as to obtain a feasible solution to \( \beta \)-FCFS;
3. computing the optimal multiplier, i.e., solving the lagrangian dual problem;

The solutions quality found for \( \beta \)-FCFS by our implementation and the computational times are discussed in Section 2.3.

**Solving the lagrangian relaxation**

Let \( L_1(k,t) \) and \( L_2(k,t) \) respectively denote the instances of problems (2.6) and (2.7) constructed by taking the first \( t-k \) rows and \( t \) columns of \( A = \{a_i(t)\}_{i \in A, t \in T} \) and the first \( k \) rows and \( t \) columns of \( B = \{b_i(t)\}_{i \in B, t \in T} \).

For any given \( \lambda \geq 0 \), \( k = 1, \ldots, n_2 \), \( t = 1, \ldots, m \) denote as \( L_i(\lambda,k,t) (i = 1,2) \) the optimal value for \( L_i(k,t) \) (notice that by Definition 2.1.2-(ii) such a value is not defined for \( k > t \)).

Let us first focus on problem (2.6).

First observe that for \( \lambda > 1 \) one clearly has

\[ L_1(\lambda) = \lambda \sum_{j=1}^{\tilde{m}} b_j(j) \]

where \( \tilde{m} = \min\{n_2, m\} \). To compute \( L_1(\lambda) \) for \( 0 \leq \lambda \leq 1 \), associate \( L_1(\lambda,k,t) \) with a node \((k,t)\) of a square grid, \( k = 0,1,\ldots,\tilde{m}, \ t = 0,1,\ldots,m \). The horizontal edges of the grid are of the form \((k-1,t-1) \rightarrow (k,t)\) and are weighted by \( \lambda b_k(t) \); the vertical ones are of the form \((k,t-1) \rightarrow (k,t)\) and are weighted by \((1-\lambda)a_{t-k}(t)\) (see Figure 2.1, where \( s = m - \tilde{m} \)).
Based on this construction one can easily prove the following

**Theorem 2.2.1.** For any $\lambda \geq 0$

\[
L_1(\lambda, k, t) = \max \left\{ L_1(\lambda, k, t - 1) + (1 - \lambda)a_{t-k}(t); \right.
\]

\[
\left. L_1(\lambda, k-1, t-1) + \lambda b_k(t) \right\}
\]

for $k = 1, \ldots, m, t = 1, \ldots, m, k \leq t$ where

\[
L_1(\lambda, 0, 0) = 0
\]

\[
L_1(\lambda, 0, t) = (1 - \lambda) \sum_{i=1}^{t} a_i(t) \quad \text{for } t = 1, \ldots, m
\]
Proof. By Definitions 2.1.1-2.1.2, the schedules in $S$ are in a one-to-one correspondence with the directed paths of the grid originating in node $(0,0)$ (such as the bold one in Figure 2.1): in particular, if edge $(k-1,t-1) \rightarrow (k,t)$ (edge $(k,t-1) \rightarrow (k,t)$) belongs to the path, then the $k$-th job of $B$ (the $(t-k)$-th job of $A$) is assigned to slot $t$. Thus a schedule maximizing $(1-\lambda)a(\sigma)+\lambda b(\sigma)$ corresponds to a path with maximum weight: the thesis follows. \hfill $\Box$

Theorem 2.2.1 allows to solve problem (2.6) in $O(\tilde{m}m)$ time. Problem (2.7) can be solved in a similar way and within the same time bound. It is sufficient to set $f_{kt}(\lambda) = L_2(\lambda,k,t) - \lambda \beta$ and give to the vertical edges of the grid having the form $(k,t-1) \rightarrow (k,t)$ a weight equal to $a_{t-k}(t)$. Then one has

$$ f_{kt}(\lambda) = \max\{f_{k,t-1}(\lambda) + a_{t-k}(t); f_{k-1,t-1}(\lambda) + \lambda b_k(t)\} \quad (2.10) $$

for $k = 1, \ldots, m$, $t = 1, \ldots, m$, $k \leq t$ where

$$ f_{00}(\lambda) = 0 \quad (2.11) $$

and

$$ f_{0t}(\lambda) = \sum_{i=1}^{t} a_i(i) $$

for $t = 1, \ldots, m$.

**Observation 2.2.2.** The above method can be extended to $u > 2$ users. In general, for $u$ users the time complexity of the algorithm is $O(m^{u-1}m)$. For an arbitrary number of users, problems (2.6)-(2.7) polynomially reduce from $1|\text{chains}; p_j = 1|\sum T_j$ and are therefore NP-hard [12].

Let now $S(\lambda)$ be the ordered set of nodes touched by a maximum path of the grid from node $(0,0)$ to node $(m,m)$. (Note that the $k$-th node of $S(\lambda)$ has the form $(i,k-1)$.) The following property of maximum paths can be used in both problems to fasten the search of the lagrangian dual solution, see §2.2.2.

**Theorem 2.2.2.** Suppose $0 \leq \lambda_1 < \lambda_2$, and let $(p,h)$ and $(q,h)$ be the $(h+1)$-st nodes of $S(\lambda_1)$ and $S(\lambda_2)$, respectively. Then $p \leq q$.

*Proof.* Let $\pi_i$ denote the maximum path associated with $\lambda_i$. Indirectly, assume that there exists a node $(i,r) \in S(\lambda_1) \cap S(\lambda_2)$ such that $(i,r) \rightarrow (i+1,r+1) \in \pi_1$ and $(i,r) \rightarrow (i,r+1) \in \pi_2$. Let $(j,s)$ be the first node $\geq (i,r)$ in $S(\lambda_1) \cap S(\lambda_2)$: such a node certainly exists since both paths end in $(m,m)$. Thus, $(j,s-1) \rightarrow (j,s) \in \pi_1$ and $(j-1,s-1) \rightarrow (j,s) \in \pi_2$. 


Let $\pi_1'$ and $\pi_2'$ denote the subpaths of $\pi_1$ and $\pi_2$ from node $(i, r)$ to node $(j, s)$, and call $a_i$ (call $\lambda_i b_i$) the total weight of the vertical (horizontal) edges of $\pi_i'$. Since the horizontal (vertical) edges of $\pi_1'$ lie all above (on the right of) the horizontal (vertical) edges of $\pi_2'$, and both $a_j(t)$ and $b_j(t)$ are regular non-increasing, one has

$$a_1 \leq a_2, \quad b_1 \geq b_2$$

Referring to problem (2.6) (to problem (2.7)), the Bellman’s condition and the optimality of $\pi_1$ for $\lambda = \lambda_1$ imply

$$(1 - \lambda_1) a_1 + \lambda_1 b_1 \geq (1 - \lambda_1) a_2 + \lambda_1 b_2$$

(implies $a_1 + \lambda_1 b_1 \geq a_2 + \lambda_1 b_2$)

that is,

$$\frac{\lambda_1}{1 - \lambda_1} \geq \frac{a_2 - a_1}{b_1 - b_2}$$

$$\left( \lambda_1 \geq \frac{a_2 - a_1}{b_1 - b_2} \right)$$

Similarly, for $\pi_2$ and $\lambda = \lambda_2$

$$\frac{\lambda_2}{1 - \lambda_2} \leq \frac{a_2 - a_1}{b_1 - b_2}$$

$$\left( \lambda_2 \leq \frac{a_2 - a_1}{b_1 - b_2} \right)$$

contradicting $\lambda_1 < \lambda_2$. 

Crossover to feasible solutions of $\beta$-FCFS

A solution $\sigma$ to the lagrangian relaxation (2.7) of $\beta$-FCFS corresponding to a fixed $\lambda$ may violate the constraint $b(\sigma) \geq \beta$. The mechanism used in our implementation in order to convert $\sigma$ into a feasible solution to $\beta$-FCFS is here addressed.

For $k = 1, \ldots, l$, let $R_k$ ($S_k$) denote the $k$-th maximal subset of consecutive time slots assigned by $\sigma$ to $A$ (to $B$), and $A_k$ ($B_k$) the corresponding subset of consecutive jobs, where

- every element of $R_k$ precedes every element of $S_k$, for $k = 1, \ldots, l$;
- every element of $R_h$ (respectively, $S_h$, $A_h$, $B_h$) precedes every element of $R_k$ (respectively, $S_k$, $A_k$, $B_k$), for $h < k$;

and $R_1$ or $S_l$ (or both) may be empty. Observe that the elements of $A_k \times R_k$ and $B_k \times S_k$ define square submatrices of $A$ and $B$ whose diagonal entries, summed up over $i$, respectively give $a(\sigma)$ and $b(\sigma)$.

Let $A_k = (i + t_k, \ldots, i + t_p)$, $B_k = (j + t_p + 1, \ldots, j + t_q)$ be the jobs of $A$ and $B$ assigned to the consecutive sets $R_k = (t_k, \ldots, t_p)$, $S_k = (t_p + 1, \ldots, t_q)$, respectively ($i, j \geq 0$). Let us give the following definition:
2.2. SOLUTION ALGORITHMS

**Definition 2.2.1.** A swap consists of assigning the first job of $B_k$ (i.e., $j + t_p + 1$) to some slot $h \in R_k$, and job $i + h \in A_k$ to the first slot of $S_k$ (i.e., $t_p + 1$).

For $h = t_k$, a swap results therefore in moving $t_k$ from $R_k$ to $S_{k-1}$, and $t_p + 1$ from $S_k$ to $R_k$. For $h \neq t_k$, the outcome is that of splitting $R_k$, $A_k$ and inserting new (singleton) subsets $S_k$, $B_k$ between the two parts obtained. Hence, after this operation the number of subsets $R_k$, $S_k$, $A_k$ and $B_k$ is either unchanged or increased by 1.

By construction, any solution obtained by swap from a feasible solution to problem (2.7) remains feasible. Moreover, being $a_j(t)$ and $b_j(t)$ regular, after a swap $a(\sigma)$ is reduced by $|\Delta a|$ and $b(\sigma)$ is increased by $|\Delta b|$ (for the sake of presentation the formulæ of $|\Delta a|$ and $|\Delta b|$ are omitted).

In our implementation, a feasible solution to $\beta$-FCFS is obtained by a sequence of greedily chosen swaps ordered by non-decreasing $|\Delta a| |\Delta b|$.

**Solving the lagrangian dual**

Theorem 2.2.2 allows to carry out the computation of the multiplier $\lambda^*$ minimizing $L_i(\lambda)$ by a standard binary search. This technique improves the overall algorithm efficiency: in fact solving the lagrangian relaxation requires lesser and lesser time as the search interval is narrowed, since by Theorem 2.2.2 the number of states enumerated by the recursive search reduces at each subgradient iteration. In our application this is a key issue to cope with the real-time requirements.

As observed in §2.2.2, for M-FCFS one has $\lambda^* \in [0, 1]$. For $\beta$-FCFS we can prove the following

**Theorem 2.2.3.** An upper bound $\lambda_{\text{max}}$ to the optimal multiplier for $\beta$-FCFS can be computed in $O(\tilde{\mu})$ time.

**Proof.** Let $\sigma_{\text{max}}$ be an optimal solution to

$$\max_{\sigma \in S}\{a(\sigma) \mid b(\sigma) = \sum_{i=1}^{\tilde{\mu}} b_i(i)\}$$

Since $L_2(\lambda)$ is piecewise linear, we can assume $\lambda_{\text{max}}$ greater than or equal to the leftmost $\lambda$ for which

$$L_2(\lambda) = (b(\sigma_{\text{max}}) - \beta)\lambda + a(\sigma_{\text{max}})$$

Let $t_i = \min\{\sigma_{\text{max}}(i) + 1, \sigma_{\text{max}}(i + 1) - 1\}$. Then for $\lambda \leq \lambda_{\text{max}}$

$$L_2(\lambda) \leq (b^* - \beta)\lambda + a^*$$
where
\[ b^* = b(\sigma_{\text{max}}) - \min_{1 \leq i \leq m} \{ b_i(\sigma_{\text{max}}(i) - b_i(t_i)) \} \]
\[ a^* = \sum_{i \in A} a_i(i) \]

Hence
\[ \lambda_{\text{max}} \leq \frac{a^* - a(\sigma_{\text{max}})}{b(\sigma_{\text{max}}) - b^*} \]

and, as the computation of the latter bound reduces to that of \( \sigma_{\text{max}} \), the thesis follows. \( \square \)

### 2.2.3 An iterative greedy-like heuristic

Scheduling problems with strict real-time constraints are generally solved by list scheduling heuristics, see \[10\]. The structure of the problems considered allows however a slightly more sophisticated approach.

Let \( \bar{m} = \min\{n_1 + n_2, m\} \) and \( \lambda \in \mathbb{R}_+ \). The following O(\( \bar{m} \)) greedy-like heuristic yields a feasible solution to \( \beta \)-FCFS, provided that \( \lambda \) is sufficiently large and such a solution exists.

1. \( t := 1 \), \( i := 1 \), \( j := 1 \) (\( i \) and \( j \) respectively denote the job of \( A \) and \( B \) candidate to be assigned to the current time slot \( t \in T \));
2. if \( \lambda b_j(t) \geq a_i(t) \), then \( \sigma(j) := t \) and \( j := j + 1 \) (that is, \( t \) is assigned to \( j \in B \)); otherwise \( \sigma(i) := t \) and \( i := i + 1 \) (that is, \( t \) is assigned to \( i \in A \));
3. if \( j > n_2 \) (\( i > n_1 \)) then assign the remaining slots to \( A \) (to \( B \)) and stop.
4. \( t := t + 1 \); if \( t \leq m \), then go to step 2, otherwise stop.

**Theorem 2.2.4.** Let \( \sigma_1 \) and \( \sigma_2 \) be the output of the above heuristic for \( \lambda_1 \) and \( \lambda_2 \). Then \( \lambda_1 \leq \lambda_2 \) implies \( a(\sigma_1) \geq a(\sigma_2) \) and \( b(\sigma_1) \leq b(\sigma_2) \).

**Proof.** By the regularity of \( a_j(t) \) and \( b_j(t) \), to prove \( a(\sigma_1) \geq a(\sigma_2) \) it is sufficient to show that for any \( i \in A \), \( \sigma_1(i) \leq \sigma_2(i) \). Indirectly, let \( i \) be the first job of \( A \) such that \( s = \sigma_1(i) > \sigma_2(i) \). Suppose that at the \( t \)-th iteration of the greedy-like heuristic with parameter \( \lambda_2 \) the incumbent jobs are \( i \in A \) and some \( j \in B \), and let \( a_i(t) > \lambda_2 b_j(t) \)

that is, \( \sigma_2(i) = t \). Since \( \sigma_2 \) is FCFS, at time \( t \) job \( i \) is preceded by the first \( (i - 1) \) jobs of \( A \) and the first \( (j - 1) \) jobs of \( B \) (therefore \( t = i + j - 1 \)). Similarly, in \( \sigma_1 \) job \( i \in A \) is preceded by the first \( (i - 1) \) jobs of \( A \) and the first \( (j - 1) + (s - t) \) jobs of \( B \). Since \( \sigma_1(k) \leq \sigma_2(k) \) for all \( k \in A \) with \( k < i \), at
the \( t \)-th iteration of the greedy-like heuristic with parameter \( \lambda_2 \) the incumbent jobs are \( i \in A \) and \( j \in B \), and
\[
a_i(t) \leq \lambda_1 b_j(t)
\]
i.e. \( \lambda_1 > \lambda_2 \) (contradiction). A similar argument proves the second inequality.

By Theorem 2.2.4, the greedy-like heuristic can be applied within a dichotomic search in the interval \([\lambda_{\text{min}}, \lambda_{\text{max}}]\) seeking for the maximum value of \( a(\sigma) \). For M-FCFS, one can set \( \lambda_{\text{min}} = 0, \lambda_{\text{max}} = 1 \).

For \( \beta \)-FCFS it is readily seen that the significant values of \( \lambda \) belong to
\[
\Lambda = \left\{ \frac{a_i(t)}{b_j(t)} \mid i \in A, j \in B, t \in T \right\}
\]
and therefore one can set \( \lambda_{\text{max}} = \max\{\lambda \mid \lambda \in \Lambda\} \). A tighter value for \( \lambda_{\text{max}} \) can however be obtained by observing that, among all the schedules \( \sigma \) returned by the heuristic, the minimum \( a(\sigma) \) corresponds to \( \hat{\sigma}(j) = j \) for \( j \in B, j \leq m \) and \( \hat{\sigma}(i) = n_2 + i \) for \( i \in A, 0 < i \leq n - m \). It is easy to see that the heuristic returns \( \hat{\sigma} \) for \( \lambda_{\text{max}} \) corresponding to the maximum in
\[
\tilde{\Lambda} = \left\{ \frac{a_1(t)}{b_j(t)} \mid j \in B, t \in T \right\}
\]
(Note that \( \lambda_{\text{max}} \) can be computed when reading the \( a_i(t) \)'s and \( b_j(t) \)'s, thus its computation does not affect the heuristic asymptotic complexity.)

### 2.3 Computational experience

The computational experience has two major purposes: (i) evaluating the quality of the solutions obtained by the lagrangian heuristic (LH) described in §2.2.2 and by the iterative greedy-like heuristic (IGH) described in §2.2.3, and (ii) checking whether the computational times of heuristics are compatible or not with the real-time constraint.

In the proposed experiments a time horizon of 3.36 seconds, i.e., 168 TTI, is considered. The experience refers to the solution of \( \beta \)-FCFS (Problem 2.1.2). A fairness criterion is used to establish the threshold value \( \beta \), which is fixed to half the maximum overall profit reachable by \( B \) in absence of \( A \). Typical features of the FUTURE test bed are summarized in Table 2.1, where the acronym PDU, standing for Packet Data Unit, indicates the standard unit obtained by the Short-term Scheduler after packet segmentation and padding (see §1.1). Constant within a block, the PDU size may vary from block to block depending on the chosen transport format (see [5]).
service typology  voice, ftp
PDU size for voice (bit)  560, 960
PDU size for ftp (bit)   1200, 1530, 2030

Table 2.1: test bed features.

Thirty instances of $\beta$-FCFS are here considered, with the characteristics listed in Table 2.2. For each user $A, B$ we report the number of PDUs, their type (ftp, $m$ = mixed (ftp and voice), $v$ = voice) and the number of data blocks obtained after segmentation.

The algorithms were implemented in language 'C', compiler Visual C/C++ 6.0, and the experiments carried out on a PC Pentium III 900 Mhz with 768 Mb RAM.

The results are reported in Table 2.3 which contains, for each instance, the fixed threshold $\beta$, the best solution $z_{L,H}$ found by algorithm LH, the optimal solution $L_2(\lambda^*)$ of the lagrangian dual problem, the percentage gap $\frac{L_2(\lambda^*)-z_{L,H}}{L_2(\lambda^*)}$, the time spent by LH to solve the lagrangian dual, the time $t^*$ spent by LH to find the best feasible solution, the best solution $z_{IGH}$ found by algorithm IGH, the percentage gap $\frac{L_2(\lambda^*)-z_{IGH}}{L_2(\lambda^*)}$ and the computation time for IGH.

Let us begin with discussing the quality of the solutions obtained by LH and IGH. Under this respect, LH outperforms IGH in all instances. In two third of the instances tested (remarkably, $I_4, I_9, I_{10}, I_{12}, I_{21}, I_{22}, I_{24}, I_{28}, I_{30}$), the advantage of LH is quite relevant. In seven of such instances, IGH fails in finding a feasible solution with a positive profit for user $A$. In eleven instances, $\%\text{gap}_{L,H}$ is smaller than 2%, and is very unsatisfactory (i.e., over 50%) in two instances only ($I_8, I_{15}$, where the performance of IGH is even worse).

Regarding the computation times we observe what follows. A time limit $T_1 = 150$ msec imposed by voice (the most popular and time stringent application) was considered. A weaker time limit, often considered satisfactory in multi-application settings, is $T_2 = 10\%$ of the planning horizon, i.e., 0.336 seconds.

The computation of the optimal value of the lagrangian dual by LH exceeds $T_1$ in all instances but one ($I_{13}$), can be carried out within $T_2$ in eleven cases, and in the worst case ($I_{21}$) takes 593 msec. Nevertheless, looking at the $t^*$ column, one observes that the best feasible solution found by LH is always computed largely within $T_1$, except two cases: $I_{18}$ and $I_{21}$.

The behaviour of IGH is more robust. In fact, IGH finds a feasible solution to $\beta$-FCFS within $T_1$ in all instances but one ($I_7$). Nevertheless, in twenty five cases (83.3% of the instances) IGH takes more time than that required by LH.
2.3. COMPUTATIONAL EXPERIENCE

<table>
<thead>
<tr>
<th>Name</th>
<th>#PDUs</th>
<th>packet type</th>
<th>#blocks</th>
<th>#PDUs</th>
<th>packet type</th>
<th>#blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>I1</td>
<td>148</td>
<td>m</td>
<td>97</td>
<td>130</td>
<td>m</td>
<td>74</td>
</tr>
<tr>
<td>I2</td>
<td>130</td>
<td>m</td>
<td>74</td>
<td>148</td>
<td>m</td>
<td>97</td>
</tr>
<tr>
<td>I3</td>
<td>196</td>
<td>m</td>
<td>75</td>
<td>273</td>
<td>m</td>
<td>169</td>
</tr>
<tr>
<td>I4</td>
<td>196</td>
<td>m</td>
<td>75</td>
<td>196</td>
<td>m</td>
<td>75</td>
</tr>
<tr>
<td>I5</td>
<td>170</td>
<td>ftp</td>
<td>150</td>
<td>196</td>
<td>m</td>
<td>75</td>
</tr>
<tr>
<td>I6</td>
<td>273</td>
<td>m</td>
<td>169</td>
<td>170</td>
<td>ftp</td>
<td>170</td>
</tr>
<tr>
<td>I7</td>
<td>437</td>
<td>m</td>
<td>310</td>
<td>273</td>
<td>m</td>
<td>169</td>
</tr>
<tr>
<td>I8</td>
<td>262</td>
<td>v</td>
<td>131</td>
<td>273</td>
<td>m</td>
<td>169</td>
</tr>
<tr>
<td>I9</td>
<td>145</td>
<td>m</td>
<td>129</td>
<td>262</td>
<td>v</td>
<td>131</td>
</tr>
<tr>
<td>I10</td>
<td>117</td>
<td>v</td>
<td>60</td>
<td>262</td>
<td>v</td>
<td>131</td>
</tr>
<tr>
<td>I11</td>
<td>437</td>
<td>m</td>
<td>310</td>
<td>117</td>
<td>v</td>
<td>60</td>
</tr>
<tr>
<td>I12</td>
<td>170</td>
<td>ftp</td>
<td>150</td>
<td>262</td>
<td>v</td>
<td>131</td>
</tr>
<tr>
<td>I13</td>
<td>257</td>
<td>m</td>
<td>102</td>
<td>117</td>
<td>v</td>
<td>43</td>
</tr>
<tr>
<td>I14</td>
<td>257</td>
<td>m</td>
<td>159</td>
<td>145</td>
<td>m</td>
<td>129</td>
</tr>
<tr>
<td>I15</td>
<td>273</td>
<td>m</td>
<td>121</td>
<td>273</td>
<td>m</td>
<td>121</td>
</tr>
<tr>
<td>I16</td>
<td>262</td>
<td>v</td>
<td>89</td>
<td>145</td>
<td>m</td>
<td>129</td>
</tr>
<tr>
<td>I17</td>
<td>145</td>
<td>m</td>
<td>129</td>
<td>273</td>
<td>m</td>
<td>121</td>
</tr>
<tr>
<td>I18</td>
<td>148</td>
<td>m</td>
<td>97</td>
<td>170</td>
<td>ftp</td>
<td>150</td>
</tr>
<tr>
<td>I19</td>
<td>130</td>
<td>m</td>
<td>74</td>
<td>262</td>
<td>v</td>
<td>89</td>
</tr>
<tr>
<td>I20</td>
<td>148</td>
<td>m</td>
<td>105</td>
<td>130</td>
<td>m</td>
<td>99</td>
</tr>
<tr>
<td>I21</td>
<td>196</td>
<td>m</td>
<td>75</td>
<td>437</td>
<td>m</td>
<td>289</td>
</tr>
<tr>
<td>I22</td>
<td>170</td>
<td>ftp</td>
<td>105</td>
<td>170</td>
<td>ftp</td>
<td>170</td>
</tr>
<tr>
<td>I23</td>
<td>262</td>
<td>v</td>
<td>131</td>
<td>170</td>
<td>ftp</td>
<td>105</td>
</tr>
<tr>
<td>I24</td>
<td>257</td>
<td>v</td>
<td>129</td>
<td>262</td>
<td>v</td>
<td>131</td>
</tr>
<tr>
<td>I25</td>
<td>170</td>
<td>ftp</td>
<td>126</td>
<td>170</td>
<td>ftp</td>
<td>100</td>
</tr>
<tr>
<td>I26</td>
<td>196</td>
<td>m</td>
<td>75</td>
<td>273</td>
<td>m</td>
<td>86</td>
</tr>
<tr>
<td>I27</td>
<td>148</td>
<td>m</td>
<td>97</td>
<td>257</td>
<td>v</td>
<td>129</td>
</tr>
<tr>
<td>I28</td>
<td>257</td>
<td>v</td>
<td>124</td>
<td>262</td>
<td>v</td>
<td>89</td>
</tr>
<tr>
<td>I29</td>
<td>170</td>
<td>ftp</td>
<td>170</td>
<td>162</td>
<td>m</td>
<td>84</td>
</tr>
<tr>
<td>I30</td>
<td>145</td>
<td>m</td>
<td>129</td>
<td>170</td>
<td>ftp</td>
<td>150</td>
</tr>
</tbody>
</table>

Table 2.2: instance characteristics.

to find its best feasible solution.

This well behaviour of LH is motivated by: (i) the multiplier adjustment guided by the lagrangian relaxation rapidly converges to values yielding (through the algorithm of Section 2.2.2) better feasible solutions than the corresponding mechanism used by IGH; (ii) Theorem 2.2.2 allows to speed up the solution of the lagrangian relaxation via a dichotomic search.

Based on this discussion, LH looks promising for application in real time sytem, though its implementation would require an additional effort concerning the development of suitable protocols and hardware supports.
### Table 2.3: computational results.

<table>
<thead>
<tr>
<th>Name</th>
<th>$\beta$</th>
<th>$z_LH$</th>
<th>$L(\lambda^*)$</th>
<th>%gap$_{LH}$</th>
<th>time$_{LH}$ (msec)</th>
<th>$t^*$</th>
<th>$z_{IGH}$</th>
<th>%gap$_{IGH}$</th>
<th>time$_{IGH}$ (msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_1$</td>
<td>75965</td>
<td>189220</td>
<td>189597</td>
<td>1.9</td>
<td>210</td>
<td>6</td>
<td>189220</td>
<td>1.9</td>
<td>30</td>
</tr>
<tr>
<td>$I_2$</td>
<td>94610</td>
<td>151930</td>
<td>152707</td>
<td>0.5</td>
<td>260</td>
<td>12</td>
<td>144730</td>
<td>5.2</td>
<td>18</td>
</tr>
<tr>
<td>$I_3$</td>
<td>148990</td>
<td>117020</td>
<td>126835</td>
<td>7.7</td>
<td>461</td>
<td>100</td>
<td>92770</td>
<td>26.8</td>
<td>55</td>
</tr>
<tr>
<td>$I_4$</td>
<td>73145</td>
<td>144370</td>
<td>145778</td>
<td>0.9</td>
<td>180</td>
<td>12</td>
<td>13440</td>
<td>90.0</td>
<td>18</td>
</tr>
<tr>
<td>$I_5$</td>
<td>73145</td>
<td>244150</td>
<td>252810</td>
<td>3.03</td>
<td>281</td>
<td>12</td>
<td>225880</td>
<td>10.2</td>
<td>25</td>
</tr>
<tr>
<td>$I_6$</td>
<td>136645</td>
<td>109440</td>
<td>161129</td>
<td>32.0</td>
<td>511</td>
<td>25</td>
<td>0</td>
<td>100</td>
<td>80</td>
</tr>
<tr>
<td>$I_7$</td>
<td>148890</td>
<td>71290</td>
<td>71763</td>
<td>0.6</td>
<td>451</td>
<td>18</td>
<td>71290</td>
<td>0.6</td>
<td>180</td>
</tr>
<tr>
<td>$I_8$</td>
<td>148890</td>
<td>72960</td>
<td>154262</td>
<td>52.7</td>
<td>441</td>
<td>18</td>
<td>0</td>
<td>100</td>
<td>65</td>
</tr>
<tr>
<td>$I_9$</td>
<td>125760</td>
<td>175380</td>
<td>183304</td>
<td>4.3</td>
<td>331</td>
<td>43</td>
<td>71490</td>
<td>60.9</td>
<td>31</td>
</tr>
<tr>
<td>$I_{10}$</td>
<td>125760</td>
<td>114460</td>
<td>134860</td>
<td>15.1</td>
<td>332</td>
<td>12</td>
<td>0</td>
<td>100</td>
<td>25</td>
</tr>
<tr>
<td>$I_{11}$</td>
<td>57230</td>
<td>71290</td>
<td>71594</td>
<td>0.4</td>
<td>193</td>
<td>6</td>
<td>56120</td>
<td>21.6</td>
<td>80</td>
</tr>
<tr>
<td>$I_{12}$</td>
<td>125760</td>
<td>186770</td>
<td>196852</td>
<td>5.1</td>
<td>375</td>
<td>19</td>
<td>67260</td>
<td>65.8</td>
<td>63</td>
</tr>
<tr>
<td>$I_{13}$</td>
<td>43160</td>
<td>233520</td>
<td>233815</td>
<td>0.1</td>
<td>131</td>
<td>6</td>
<td>233520</td>
<td>0.1</td>
<td>10</td>
</tr>
<tr>
<td>$I_{14}$</td>
<td>108345</td>
<td>206360</td>
<td>227741</td>
<td>9.3</td>
<td>418</td>
<td>43</td>
<td>162480</td>
<td>28.6</td>
<td>90</td>
</tr>
<tr>
<td>$I_{15}$</td>
<td>129480</td>
<td>101240</td>
<td>230815</td>
<td>56.1</td>
<td>356</td>
<td>68</td>
<td>98560</td>
<td>57.2</td>
<td>70</td>
</tr>
<tr>
<td>$I_{16}$</td>
<td>108345</td>
<td>183920</td>
<td>184160</td>
<td>0.1</td>
<td>518</td>
<td>18</td>
<td>183920</td>
<td>0.1</td>
<td>10</td>
</tr>
<tr>
<td>$I_{17}$</td>
<td>129480</td>
<td>184560</td>
<td>230815</td>
<td>20.03</td>
<td>406</td>
<td>13</td>
<td>95210</td>
<td>58.7</td>
<td>19</td>
</tr>
<tr>
<td>$I_{18}$</td>
<td>144920</td>
<td>149620</td>
<td>170978</td>
<td>12.4</td>
<td>531</td>
<td>362</td>
<td>8120</td>
<td>94.5</td>
<td>140</td>
</tr>
<tr>
<td>$I_{19}$</td>
<td>91960</td>
<td>151930</td>
<td>152577</td>
<td>0.4</td>
<td>293</td>
<td>12</td>
<td>151930</td>
<td>0.4</td>
<td>30</td>
</tr>
<tr>
<td>$I_{20}$</td>
<td>88045</td>
<td>202020</td>
<td>202136</td>
<td>0.05</td>
<td>343</td>
<td>12</td>
<td>202020</td>
<td>0.05</td>
<td>80</td>
</tr>
<tr>
<td>$I_{21}$</td>
<td>68820</td>
<td>132190</td>
<td>140827</td>
<td>6.1</td>
<td>593</td>
<td>300</td>
<td>0</td>
<td>100</td>
<td>120</td>
</tr>
<tr>
<td>$I_{22}$</td>
<td>136645</td>
<td>182960</td>
<td>190500</td>
<td>3.95</td>
<td>562</td>
<td>13</td>
<td>16800</td>
<td>91.1</td>
<td>50</td>
</tr>
<tr>
<td>$I_{23}$</td>
<td>116530</td>
<td>142080</td>
<td>227264</td>
<td>37.4</td>
<td>343</td>
<td>31</td>
<td>0</td>
<td>100</td>
<td>90</td>
</tr>
<tr>
<td>$I_{24}$</td>
<td>125760</td>
<td>195840</td>
<td>197053</td>
<td>0.6</td>
<td>437</td>
<td>37</td>
<td>0</td>
<td>100</td>
<td>80</td>
</tr>
<tr>
<td>$I_{25}$</td>
<td>114745</td>
<td>256000</td>
<td>263104</td>
<td>1.33</td>
<td>363</td>
<td>13</td>
<td>151080</td>
<td>42.5</td>
<td>44</td>
</tr>
<tr>
<td>$I_{26}$</td>
<td>98200</td>
<td>150520</td>
<td>151017</td>
<td>0.3</td>
<td>250</td>
<td>18</td>
<td>148600</td>
<td>1.6</td>
<td>20</td>
</tr>
<tr>
<td>$I_{27}$</td>
<td>124360</td>
<td>105380</td>
<td>158903</td>
<td>33.6</td>
<td>500</td>
<td>43</td>
<td>8120</td>
<td>94.8</td>
<td>110</td>
</tr>
<tr>
<td>$I_{28}$</td>
<td>91960</td>
<td>238160</td>
<td>238286</td>
<td>0.05</td>
<td>312</td>
<td>18</td>
<td>5760</td>
<td>97.5</td>
<td>60</td>
</tr>
<tr>
<td>$I_{29}$</td>
<td>76520</td>
<td>253750</td>
<td>259575</td>
<td>2.24</td>
<td>475</td>
<td>6</td>
<td>170520</td>
<td>94.3</td>
<td>44</td>
</tr>
<tr>
<td>$I_{30}$</td>
<td>144920</td>
<td>149370</td>
<td>154365</td>
<td>6.11</td>
<td>531</td>
<td>13</td>
<td>0</td>
<td>100</td>
<td>56</td>
</tr>
</tbody>
</table>
Chapter 3

Polyhedral study of one-machine scheduling problem: the case of a single user

3.1 Introduction

In integer programming, a typical approach in order to study a general problem consists in finding a linear inequality description of the set of feasible points. Given a discrete optimization problem, \(\min\{cx : x \in S\}\) where \(S \subseteq \mathbb{Z}_+^n\), let us define \(P = \{x \in \mathbb{R}_+^n : Ax \leq b\}\), where \(A \in \mathbb{R}^{m \times n}\) and \(b \in \mathbb{R}^m\). We say that \(P\) is a formulation of the original problem if and only if \(P \cap \mathbb{Z}_+^n = S\). In general, it is possible to have infinite formulations associated with the same problem and could be very important to determine the best among them.

One possible criterion to establish if formulation \(P_1\) is better than \(P_2\) consists in solving the linear relaxation of the original problem whose solution is a bound for the optimum (in particular, if we minimize the objective function the solution is a lower bound). Solving the linear relaxation, we calculate the value \(LB(P)\); on the other hand, we can implement some heuristic to find a feasible solution \(\bar{x}\) of the integer problem. The gap (\(\bar{x} - LB(P)\)) certifies the quality of the solution. Obviously, the gap depends on the formulation; since a small gap implies a good solution, we are interested in formulations associated with a large lower bound. Then, if the lower bound associated with \(P_1\) is greater than the lower bound associated with \(P_2\), we conclude that \(P_1\) is better than \(P_2\). However, we can immediately see that this criterion depends on the objective function which is not known a priori.

In alternative way, we say that \(P_1\) is better than \(P_2\) if and only if \(P_1 \subseteq P_2\). The
set of all points that are convex combinations of points in $S$, i.e. the \textit{convex hull} of $S$, $\text{conv}(S)$, corresponds to the only formulation contained in all the other possible formulations of the original problem. Then, the objective of the polyhedral study consists in completely describing the convex hull associated with the current problem: it makes to solve the linear programming problem $\min\{cx : x \in \text{conv}(S)\}$ which is computationally easy.

The total description of $\text{conv}(S)$ can be very hard to achieve and involves also the study of the \textit{dimension} of the convex hull. We remember that a general polyhedron $P \subseteq \mathbb{R}^n$ is of dimension $d$, $\dim(P) = d$, if the maximum number of affinely independent points in $P$ is $d + 1$; if $\dim(P) = n$, then $P$ is \textit{full-dimensional}.

To completely describe $\text{conv}(S)$ means that we have to establish which among the inequalities in $Ax \leq b$ are indeed necessary and which not for formulating the problem.

Given a general polyhedron $P$, we say that inequality $ax \leq a_0$ is \textit{valid} for $P$ if it is satisfied by all the points in $P$. Given the valid inequality $ax \leq a_0$, the set $F = \{x \in P : ax = a_0\}$ is a \textit{face} of $P$ and we say that this inequality \textit{represents} $F$. If $F \neq \emptyset$ and $F \neq P$, then $F$ is a \textit{proper face} of $P$; further, when $F \neq \emptyset$ we say that inequality $ax \leq a_0$ \textit{supports} $P$. We can count out of the description of $\text{conv}(S)$ all the inequalities that are not a support of it.

In general, for a proper face $F$ of $P$ it happens that $\dim(F) < \dim(P)$; if $\dim(F) = \dim(P) - 1$, then $F$ is a \textit{maximal face} or \textit{facet-defining inequality} of $P$. In correspondence with any facet, some inequalities exist which represent $P$ and one among them is necessary in the description of $P$. Instead, every inequality representing a face $F$ such that $\dim(F) < \dim(P) - 1$ can be removed from the description of $P$.

Then, the main task of the polyhedral study consists in deciding which among the valid inequalities of $S \subseteq \mathbb{Z}^n$ are facets of $\text{conv}(S)$. If we assume that $\text{conv}(S)$ is full-dimensional, we have two different approaches to prove this result depending on whether we use the definition or not. In the first case, we have to find $n$ points in $S$ verifying the current inequality and prove that they are affinely independent. Otherwise, we can suppose that there exist an hyperplane $\pi^T x = \pi_0$ containing $S$ and select $k \geq n$ points in $S$; since these points belong also to the hyperplane, we solve the associated systems. If the only solution implies that the hyperplane coincides with the current inequality, then the result is proved. In this thesis, we will use the latter approach in order to prove that a valid inequality is a facet.

It is possible that some inequalities are valid for lower dimensional restrictions of $S$ but not for $S$. Then, we can apply a particular procedure, called \textit{lifting}, for deriving new inequalities which are valid for $S$. Further, if they are
3.1. INTRODUCTION

also facets for some restrictions by the lifting we can try to change them in facets for $S$. However, observe that it is not always the case that inequalities valid for restrictions are valid also for the original polyhedron.

Lifting with general integer variables is computationally harder than lifting with 0-1 variables, since in the first case the lifting procedure involves the resolution of a nonlinear integer problem. Then, from now on we assume that $S \subseteq \{0,1\}^n$.

We can illustrate the lifting principle as follows:

1. Let $S^k = S \cap \{x \in \{0,1\}^n : x_1 = k\} \text{ for } k \in \{0,1\}$. Suppose that $\sum_{i=2}^n a_i x_i \leq \alpha_0$ is valid for $S^0$.

2. If $S^1 = \emptyset$, then $x_1 \leq 0$ is a valid inequality for $S$.

3. If $S^1 \neq \emptyset$, then $\alpha_1 x_1 + \sum_{i=2}^n a_i x_i \leq \alpha_0$ is valid for any $\alpha_1 \leq \alpha_0 - \gamma$ where $\gamma = \max\{\sum_{i=2}^n a_i x_i : x \in S^1\}$.

4. If $\sum_{i=2}^n a_i x_i \leq \alpha_0$ is a facet of $\text{conv}(S^0)$ and $\alpha_1 = \alpha_0 - \gamma$, then $\alpha_1 x_1 + \sum_{i=2}^n a_i x_i \leq \alpha_0$ is a facet of $\text{conv}(S)$.

The parameter $\alpha_1$ is called lifting coefficient.

In general, we can start from a valid inequality for the set $S \cap \{x \in \{0,1\}^n : x_k = 0, \text{ for some } k \in \{1, \ldots, n\}\}$ and add variables $x_k$ by different sequences each of them may provide distinct facets. We refer to this procedure as sequential lifting. Alternately, we can consider another procedure called simultaneous lifting: in this case, the coefficients of all variables that are to be lifted are simultaneously considered, yielding inequalities that cannot be obtained by sequential lifting. For more details on the lifting principle, see [14].

Once all the facets have been calculated, we can use a lot of techniques to show that they completely describe $\text{conv}(S)$. Here we present seven approaches allowing to prove that the polyhedron $P = \{x \in \{0,1\}^n : Ax \leq b\}$ describes $\text{conv}(S)$.

1. Show that the matrix $A$, or the pair $(A,b)$ have special structure guaranteeing that $P = \text{conv}(S)$.

2. Show that points $x \in P$ with fractional components are not extreme points of $\text{conv}(S)$ (we remember that $x$ is an extreme point of a general polyhedron $P$ if there do not exist $x^1$ and $x^2$ in $P$, $x^1 \neq x^2$ such that $x = \lambda x^1 + (1 - \lambda)x^2$, $\lambda \in (0,1)$).

3. Show that for all $c \in \mathbb{R}^n$, the linear program $z^{LP} = \max\{cx : Ax \leq b\}$ has an optimal solution $x^* \in S$. 


4. Show that for all $c \in \mathbb{R}^n$, there exists a point $x^* \in S$ and a feasible solution $u^*$ of the dual problem $w^{LP} = \min \{ub : uA = c, u \geq 0\}$ with $cx^* = u^*b$. Note that this implies that the condition of the approach 3 is satisfied.

5. Show that if $ax \leq a_0$ defines a facet of $\text{conv}(S)$, then it must be identical to one of the inequalities defining $P$.

6. Show that for any $c \in \mathbb{R}^n$, $c \neq 0$, the set of optimal solutions $M(c)$ to the problem $\max\{cx : x \in S\}$ lies in $\{x : a^ix = b_i\}$ for some $i = 1, \ldots, m$, where $a^ix \leq b_i$ for $i = 1, \ldots, m$ are the inequalities defining $P$.

7. A set of linear inequalities $Ax \leq b$ is called Total Dual Integral (TDI) if, for all $c \in \mathbb{Z}^n$ for which the linear program $\max\{cx : Ax \leq b\}$ has a finite optimal value, the dual linear program $\min\{yb : yA = c, y \geq 0\}$ has an optimal solution with $y$ integral. If $Ax \leq b$ is TDI, $b$ is an integer vector, and $P$ has vertices, then all vertices of $P$ are integer.

Exploiting the above definition and result, verify that $Ax \leq b$ is TDI.

In this work, we prove the argument by using the second approach in the above list.

In this second part of the thesis, we will provide a polyhedral study of the max-sum competitive scheduling problem 2.1.3 illustrated in §2.1.1. Different versions of this problem are considered, depending on the total number $n$ of users in the system and on whether a feasible schedule must or may not be complete.

Given the sets of jobs and time-slots

$$J_k = \{1, 2, \ldots, n_k\}, k = 1, \ldots, n \quad \text{and} \quad T = \{1, 2, \ldots, m\}$$

we assume $m \geq n_1 + \ldots + n_n$. Moreover, $i \rightarrow j \iff i < j$ for any $i, j \in J_k$. Since the same index can denote jobs in different $J_k$, the superscript $k$ will be used when necessary.

A time-indexed formulation is based on time-discretization. Since we consider the planning horizon $T$, we define 0-1 variables $x^k_{jt}$ for any job in $J_k$, $k = 1, \ldots, n$, and for any slot $t$ in $T$. Each variable specifies the assignment of a job to a slot: precisely, $x^k_{jt} = 1$ if and only if $j \in J_k$ is assigned to $t \in T$. The time-indexed formulation for the general case of Problem 2.1.3 is
3.1. INTRODUCTION

\[\max \sum_{k=1}^{m} \sum_{j \in J_k} \sum_{t \geq j} f^k_j(t) x^k_{jt} \]  
\[\sum_{t=j}^{n} x^k_{jt} \leq 1 \quad j \in J_k, \forall k \]  
\[\sum_{k=1}^{m} \sum_{j=1}^{t} x^k_{jt} \leq 1 \quad t \in T \]  
\[x^k_{jt} - \sum_{s=j-1}^{t-1} x^k_{j-1,s} \leq 0 \quad j \in J_k, j \neq 1, t \geq j, \forall k \]  
\[x^k_{jt} \geq 0 \quad j \in J_k, t \geq j, \forall k \]  
\[x^k_{jt} \text{ integer} \quad j \in J_k, t \geq j, \forall k \]

where \(f^k_j(t)\) is the revenue obtained by scheduling, and therefore completing, \(j \in J_k\) at time \(t\). Inequalities (3.4) represent the precedence relations among the jobs of \(J_k\): if \(j\) is not the first job of \(J_k\) and is scheduled at time \(t\), then \(j - 1\) must be scheduled at least at time \(t - 1\). Notice that a job (a time slot) does not need to be assigned to a time slot (a job): this variant of the problem is called Partial Scheduling Problem, PSP\(_n\) (the index refers to the number of chains). To require that all jobs are scheduled, one has to replace the sign ‘\(\leq\)’ with ‘\(=\)’ in constraints (3.2): this variant is here referred to as Complete Scheduling Problem, CSP\(_n\). A variant of Complete Scheduling is finally when \(n > 1\), \(m = \sum n_k\): such a variant is the Shuffling Problem, SP\(_n\).

In particular, we are interested in finding the description of the polytope defined as the convex hull of all vectors fulfilling assignment and precedence constraints and having integer components. The polyhedron will be studied in the following cases:

- \(n = 1, m \geq n_1\) (P/CSP\(_1\))
- \(n = 2, m \geq n_1 + n_2\) (PSP\(_2\))
- \(n = 2, m > n_1 + n_2\) (CSP\(_2\))
- \(n = 2, m = n_1 + n_2\) (SP\(_2\))

A dynamic programming algorithm which finds an optimal solution can be provided for several variants of the max-sum problem. We here report CSP\(_1\), CSP\(_2\) and SP\(_2\).
1. For CSP\(_1\), let \( f(j, s) \) denote the utility of an optimal solution assigning exactly \( j \) jobs of \( J_1 \) to \( s \geq j \) slots. Then

\[
f(j, s) = \max \{ f(j, s - 1); f(j - 1, s - 1) + f^1_j(s) \}
\]

under the initial conditions \( f(0, s) = 0 \) for each \( s \in T \) and \( f(j, s) = -\infty \), for all \( j \in J_1 \) and \( s < j \). The computation requires \( O(n_1 m) \) time.

2. For CSP\(_2\), consider an optimal assignment of exactly \( j \) jobs in \( J_1 \) and of exactly \( k \) jobs in \( J_2 \) to \( s \geq j+k \) slots. The corresponding utility \( f(j, k, s) \) is inductively computed as:

\[
f(j, k, s) = \max \{ f(j-1, k, s-1) + f^1_j(s); f(j, k-1, s-1) + f^2_k(s); f(j, k, s-1) \}
\]

under the initial condition \( f(0, 0, s) = 0 \) for all \( s \in T \). The computation requires \( O(n_1 n_2 m) \) time.

3. For SP\(_2\), suppose that in an optimal assignment of \( s \leq m \) slots, \( k \) slots are assigned to (the first \( k \) jobs of) \( J_2 \) and the remaining to \( J_1 \). The corresponding utility \( f(k, s) \) is inductively computed as:

\[
f(k, s) = \max \{ f(k, s - 1) + f^1_{s-k}(s); f(k - 1, s - 1) + f^2_k(s) \}
\]

under the initial condition \( f(0, 0) = 0 \). Observe that \( f(k, s) \) is defined for \( k \leq s \leq k + n_1 \) and the computation requires \( O(m^2) \) time. Formula (3.9) can be extended to P/CSP\(_2\) if \( f^k_j(t) \) is non-negative regular (i.e., non-increasing with \( t \)).

The remainder of this chapter is organized as follows: in section 3.2 we will study PSP\(_1\). We will give a time-indexed formulation of the problem, prove that the polyhedron defined as the convex hull of feasible schedules is full-dimensional and completely describe it by showing that each its vertex is integer. However, we will study some peculiar properties of the conflict graph associated with this problem. In section 3.3 we will study CSP\(_1\). We will illustrate an alternative technique to study the polyhedron associated with the convex hull based on the reformulation of the recursion procedure used to solve it. This approach can be applied since the problem is solvable in polynomial time.

### 3.2 PSP\(_1\)

In this section, we study the polyhedron when only one user is in the system. The notation is simplified as follows: \( J_1 = A, x^1_{jt} = x_{jt}, f_j(t) = a_j(t) \) for all \( j \in A \) and \( t \in T \).
The time-indexed formulation is

\[
\max \sum_{i \in A} \sum_{t \geq i} a_i(t)x_{it} \quad (3.10)
\]

\[
\sum_{t \geq i} x_{it} \leq 1 \quad i \in A \quad (3.11)
\]

\[
\sum_{t \leq i} x_{it} \leq 1 \quad t \in T \quad (3.12)
\]

\[
x_{it} - \sum_{s=i-1}^{t-1} x_{i-1,s} \leq 0 \quad i \in A, i \neq 1, \ t \geq i \quad (3.13)
\]

\[
x_{it} \geq 0 \quad i \in A, \ t \geq i \quad (3.14)
\]

\[
x_{it} \text{ integer} \quad i \in A, \ t \geq i \quad (3.15)
\]

We observe that \(x_{it}\) is defined only for \(t \geq i\).

The assignment constraints (3.11) (3.12) state that each job (slot) can be assigned at most once, and the precedence constraints (3.13) state that precedence conditions must be satisfied.

**Theorem 3.2.1.** The precedence inequalities (3.13) can be reinforced to yield the following valid inequalities

\[
\sum_{s=i-1}^{t-1} x_{i-1,s} \leq 0 \quad (3.16)
\]

for all \(i \in A, i \neq 1, \ t \geq i\).

**Proof.** Lift variables according to the following sequence \(S_1, \ldots, S_5\):

\[
S_1 = \{x_{js} : j = 1, \ldots, i - 2, s \geq j\}
\]

\[
S_2 = \{x_{is} : s = i, \ldots, t - 1\}
\]

\[
S_3 = \{x_{i-1,s} : s = t, \ldots, m\}
\]

\[
S_4 = \{x_{is} : s = t + 1, \ldots, m\}
\]

\[
S_5 = \{x_{js} : j = i + 1, \ldots, n_1, s \geq j\}
\]

Set the lifting coefficients of \(S_1, S_3, S_4, S_5\) to 0; those of \(S_2\) to 1. Observe that this procedure can be applied in the general case when \(n\) users are in the system, and then inequalities (3.16) express the precedence constraints in a general formulation.

**Observation 3.2.1.** Inequalities (3.11) are redundant for \(i \neq 1\). Similarly, inequalities \(x_{ii} \geq 0\) are redundant for \(i \neq n_1\).
Proof. Inequalities (3.11) (for \( i = 1 \)), (3.16) (for \( i = 2 \)) and (3.14) imply

\[
\sum_{s=2}^{m} x_{2s} \leq \sum_{s=1}^{m-1} x_{1s} \leq \sum_{s=1}^{m} x_{1s} \leq 1
\]

i.e., inequality (3.11) for \( i = 2 \), and so on. Moreover, inequality (3.16) for \( i = t = n_1 \) writes

\[
x_{n_1-1,n_1-1} \geq x_{n_1 n_1}
\]

Being the right-hand side \( \geq 0 \) implies \( x_{n_1-1,n_1-1} \geq 0 \), and so on. This result can be extended to the general case when \( n \) users are in the system.

\[\square\]

**Observation 3.2.2.** Inequalities (3.12) are dominated.

Proof. From (3.11) written for \( i = 1 \) and non-negativity we have

\[
1 \geq \sum_{s=1}^{m} x_{1s} \geq \sum_{s=1}^{t} x_{1s} \geq x_{1t} + \sum_{s=2}^{t} x_{2t} \geq x_{1t} + \sum_{s=2}^{t} x_{2t} \geq \ldots \geq \sum_{j=1}^{t} x_{jt}.
\]

\[\square\]

From the observations and lifting, the time-indexed formulation of PSP\(_1\) rewrites

\[
\max \sum_{i \in A} \sum_{t \geq 1} a_i(t) x_{it}
\]

\[
\sum_{t \geq 1} x_{1t} \leq 1 \quad i \in A \quad (3.17)
\]

\[
\sum_{s=i}^{t} x_{it} - \sum_{s=i-1}^{t-1} x_{i-1,s} \leq 0 \quad i \in A, i \neq 1, \ t \geq i
\]

\[
x_{it}, x_{n_1,n_1} \geq 0 \quad i \in A, \ t > i \quad (3.18)
\]

\[
x_{it} \text{ integer} \quad i \in A, \ t \geq i
\]

From now on, let \( S \) be the set of constraints (3.17), (3.16) and (3.18), and \( P_S \) the convex hull of \( S \).

**Theorem 3.2.2.** \( P_S \) is full-dimensional.
Proof. Suppose that $P_S = \{x \in P_S : \pi^T x = 0\}$. Let us show that $\pi = 0$.
Let $e_{ij}^s$ denote the $(i, j)$-th unit vector, i.e. $e_{ij}^{pq} = 1$ and $e_{pq}^{ij} = 0$ for all $(p, q) \neq (i, j)$ and define

$$x_1^s = e_1^s, \hspace{1cm} s \in T$$
$$x_i^s = \sum_{h=1}^{i-1} e_{hh}^i + e_{is}^i, \hspace{1cm} i \in A : i \geq 2, \ s \in T : s \geq i$$

Since the above vectors belong to $S$ and have integer components, it follows $\pi^T x_i^s = 0$ for all $i \in A$ and $s \in T : s \geq i$. We then obtain

$$\sum_{h=1}^{i-1} \pi_{hh} + \pi_{is} = 0 \hspace{1cm} i \in A, \ s \geq i$$

Hence, $\pi_{is} = 0$. 

In the following, we give a complete description of the inequalities defining $P_S$.

**Theorem 3.2.3.** Inequalities

$$\sum_{s \geq 1} x_{1s} \leq 1$$
$$x_{it} \geq 0 \hspace{1cm} i \in A, t > i$$
$$x_{n1n1} \geq 0$$

induce facets of $P_S$.

*Proof. See [18]*

**Theorem 3.2.4.** Inequalities (3.16) induce facets of $P_S$.

*Proof. For $i \in A, t \in T$, let

$$F_{it} = \{x \in P_S : \sum_{s=i}^{t} x_{is} = \sum_{s=i-1}^{t-1} x_{i-1,s}\}$$

Suppose that $\pi^T x = 0$ for all $x \in F_{it}$. We next show that there exists some constant $\alpha \in \mathbb{R}^+$ such that

$$\pi^T x = (\sum_{s=i}^{t} x_{is} - \sum_{s=i-1}^{t-1} x_{i-1,s}) \cdot \alpha.$$
As in Theorem 3.2.2 take
\[ x^1_s = e^1_s \quad s \in T \]
\[ x^j_s = \sum_{h=1}^{j-1} e^{hh} + e^{j-1,s} + e^{is} \quad j \in A : j \geq 2, \ s \in T : s \geq j \]

These vectors belong to \( F_{it} \), hence \( \pi^T x^j_s = 0 \), and we obtain
\[ \pi_{js} = 0 \quad \text{for} \ 1 \leq j \leq i - 2, \ s \geq j \]

Solving then the system \( \pi^T x^i_s = 0 \) for \( i \leq s \leq t \), we get
\[ \pi_{is} = -\pi_{i-1,i-1} \quad \text{for} \ i \leq s \leq t \]

Define then
\[ u^{i-1,s} = \sum_{h=1}^{i-2} e^{hh} + e^{i-1,s} + e^{it} \]
for \( i - 1 \leq s \leq t - 1 \), and solve \( \pi^T u^{i-1,s} = 0 \). Hence
\[ \pi_{i-1,s} = -\pi_{it} \quad \text{for} \ i - 1 \leq s \leq t - 1 \]

Finally, taking \( x^j_s \) for \( j > i, s \geq j \), and
\[ v^{i-1,s} = \sum_{h=1}^{i-2} e^{hh} + e^{i-1,s} \quad \forall s \in T : s \geq t \]
\[ w^s = \sum_{h=1}^{i-2} e^{hh} + e^{i-1,t} + e^{is} \quad \forall s \in T : s > t \]

and solving the associated systems, we see that the remaining components of \( \pi \) equal 0, and the thesis follows. \( \square \)

**Theorem 3.2.5.** Let \( Q \) denote the polytope defined by inequalities (3.17), (3.16) and (3.18). Then \( Q = P_S \).

**Proof.** Let \( x^* \) be a fractional vertex of \( Q \). Let us show that \( x^* \) is not an extreme point of \( Q \).

For all \( i \in A \), let
\[ t(i) = \min\{t : x^*_{it} > 0\} \]

From the precedence constraints, it follows that \( i < j \) implies \( t(i) < t(j) \). In fact
\[ 0 = \sum_{s=i}^{t(i)-1} x^*_{is} \geq \sum_{s=i+1}^{t(i)} x^*_{i+1,s} \geq \ldots \geq \sum_{s=j}^{t(i)+(j-i-1)} x^*_{js} \]
3.2. PSP

Then, each of the previous sums equals 0 (from the non-negativity constraints) and \( t(j) > t(i) + (j - i - 1) \geq t(i) \).

Let \( y \) be defined as

\[
y_{it(i)} = \lceil x_{it(i)}^* \rceil \quad y_{it} = 0 \quad \forall t \neq t(i)
\]

for \( i \in A \) and \( t \in T, t \geq i \). It is immediately seen that \( y \in Q \).

Let then \( z \) be defined as

\[
z_{it(i)} = \frac{x_{it(i)}^*}{1 - \lambda} - \frac{\lambda}{1 - \lambda} \quad z_{it} = \frac{x_{it}^*}{1 - \lambda} \quad \forall t \neq t(i)
\]

for \( i \in A, t \in T, t \geq i \) and for \( 0 < \lambda < 1, \lambda = \min\{x_{it}^*: x_{it}^* > 0\} \). Let us show that also \( z \in Q \).

i) Trivially, \( z \) verifies inequalities (3.18).

ii) \( z \) verifies inequality 3.17. In fact, since \( x^* \in Q \), one has

\[
\sum_{s=1}^{m} z_{1s} = z_{1t(1)} + \sum_{s \neq t(1)} z_{1s} = \frac{x_{1t(1)}^*}{1 - \lambda} - \frac{\lambda}{1 - \lambda} + \sum_{s \neq t(1)} \frac{x_{1s}^*}{1 - \lambda} = \frac{\sum_{s=1}^{m} x_{1s}^*}{1 - \lambda} - \frac{\lambda}{1 - \lambda} \leq 1
\]

iii) \( z \) verifies (3.16). In fact

\[
\sum_{s=i-1}^{t-1} z_{i-1,s} = \begin{cases} \frac{\sum_{s=i-1}^{t-1} x_{i-1,s}^* - \lambda}{1 - \lambda} & \text{if } t(i - 1) \leq t - 1 \quad (a) \\ 0 & \text{if } t(i - 1) > t - 1 \quad (b) \end{cases}
\]

and similarly

\[
\sum_{s=1}^{t} z_{is} = \begin{cases} \frac{\sum_{s=1}^{t} x_{i,s}^* - \lambda}{1 - \lambda} & \text{if } t(i) \leq t \quad (c) \\ 0 & \text{if } t(i) > t \quad (d) \end{cases}
\]

Using these last inequalities, (a) – (d), and the fact that if \( t(i - 1) \leq t - 1 \) then \( \sum_{s=i-1}^{t-1} x_{i-1,s}^* \geq \sum_{s=i}^{t} x_{is}^* \), one concludes that \( z \) satisfies (3.13).

Finally, we have \( x^* = \lambda y + (1 - \lambda)z \), which completes the proof. \( \square \)

We conclude this section by observing that polyhedral properties of assignment problems can also be derived from the definition of conflict graph. The conflict graph \( G = (V, E) \) associated with an instance of PSP\(_1\) is constructed as follows:

- each vertex of \( V \) is in a one-to-one correspondence with the variables \( x_{jt} \), for \( j \in A, t \geq j \);

---

**Note:** The text is simplified for readability, and some equations are presented in a more compact form. The full mathematical rigor and details are as in the original document.
two vertices in $V$ are adjacent if and only if the corresponding variables are incompatible, namely, cannot be set both to 1 in a feasible solution. The incompatibility between variables $x_{is}$ and $x_{jt}$ can be checked through the bipartite graph $B = (A \cup T, E)$, where $jt \in E$ corresponds to $x_{jt}$, and the vertices in $A$ and $T$ are sorted according to the job precedence relation and to time. Then, $x_{is}$ and $x_{jt}$ are incompatible if and only if

(a) either the corresponding edges in $A$ are adjacent,

(b) or cross each other,

(c) or there are more job-vertices between $i$ and $j$ than slot-vertices between $s$ and $t$.

**Theorem 3.2.6.** Let $G$ denote the conflict graph of $PSP_1$. Then $G$ is perfect.

**Proof.** Draw $B$ in $\mathbb{R}^2$ by associating the vertices of $A$ and $T$ with $n_1$ and $m$ equidistant points of two parallel axes $r_1, r_2$, and connecting with straight segments the points corresponding to adjacent vertices of $B$. Construct then a new bipartite graph $\tilde{B} = (A \cup \tilde{T}, \tilde{E})$ as follows. Prolong each segment $jt$ beyond $t$ until all the convergent segments intersect, and draw a third axis $r_3$ parallel to $r_1$ and crossing all the prolonged segments. Then, $\tilde{E}$ ($\tilde{T}$) is the set of all the (points of $r_3$ belonging to some) prolonged segment. An example of construction is drawn in Figure 3.1.

Clearly, two edges in $\tilde{E}$ cross each other if and only if are obtained extending two edges of $E$ for which one of (a), (b), (c) holds. Hence the conflict graph $G$ is the segment-intersection graph of $\tilde{B}$, which is weakly triangulated and therefore perfect [11].

Observe that the theorem can be extended to $PSP_n$, i.e. to the case in which $n$ users are in the system.

Now, every feasible solution to $PSP_1$ corresponds to a stable set of $G$. The converse, however, is not true: the stable set $S = \{22, 33, \ldots, n_1 n_1\}$ corresponds to $x_{22} = x_{33} = \ldots = x_{n_1 n_1} = 1$, which is not feasible because does not schedule job 1 and therefore does not fulfill the precedence constraint. In fact, feasible solutions of $PSP_1$ correspond to *maximal* stable sets of $G$. Hence, Theorem 3.2.6 allows us to transform $PSP_1$ in a maximum weighted stable set problem on $G$ for which there exists a formulation with integer optimum, provided that the utilities $a_{it}$ are $\geq 0$ for all $i \in A$ and $t \in T$. 


Figure 3.1: Constructing graph $\tilde{B}$ from $B$
3.3 CSP\textsubscript{1}

A time-indexed formulation for CSP\textsubscript{1} is obtained from (3.10)-(3.15) by defining variables \(x_{it}\) for \(i \in A\) and \(t \in T\), \(t \leq i + m - n_1\), and replacing the sign ‘\(=\)’ for ‘\(\leq\)’ in (3.11). For this reason, the convex hull of the feasible solutions of CSP\textsubscript{1} is not full-dimensional. To describe it we must therefore resort to a different technique from that used for PSP\textsubscript{1}.

This technique is based on a reformulation of the recursion (3.7) in terms of linear programming. The main steps of this method are as follows:

1. Consider the recursion formula which solves the integer linear problem.
2. By associating a variable to each value in the recursion, formulate a linear program.
3. Consider the dual of the linear program formulated in Step 2. This new problem corresponds to a reformulation of the original problem and its variables can be reinterpreted as the original 0-1 variables.
4. From the dynamic procedure, an optimal solution of the dual problem corresponds to an optimal sequence of decisions in the recursion. Then, the linear program has an optimal integer solution.

For more details about this technique, see [19].

Associate a variable \(y_{jt}\) with each value \(f(j, t)\) in formula (3.7), and define the following linear program

\[
\begin{align*}
\min & \quad y_{n_1m} \\
\text{s.t.} & \quad y_{jt} - y_{j,t-1} \geq 0 \quad \forall j \in A, \forall t > j \\
& \quad y_{1t} \geq a_{1t} \quad \forall t \geq 1 \\
& \quad y_{jt} - y_{j-1,t-1} \geq a_{jt} \quad \forall j \in A - \{1\}, \forall t \geq j
\end{align*}
\]

An optimal solution \(\mathbf{y}^*\) to (3.19) is such that \(y_{jt}^* = f(j, t)\) for \(j \in A, t \in T, t \geq j\). Let \(u_{j,t-1}, w_{1t}\) and \(w_{jt}\) be the dual variables associated with the constraints of program (3.19). The dual reads
Programs (3.19) and (3.20) have totally unimodular constraint matrices, so the extreme points of the latter polyhedron have all integer components. Moreover, solving (3.20) corresponds to finding an optimal solution to (3.10)-(3.15). To prove this, we show that the polyhedron associated with CSP\(1\) can be obtained by projecting the polyhedron defined by the constraints of program (3.20) into the subspace defined by \(u = 0\). The projection procedure is described hereafter.

**Projection 1**

1. Rewrite the first dual constraint as

\[
 w_{jj} - u_{jj} - w_{j+1,j+1} = 0 \quad \forall j \in A - \{n_1\} 
\]

for \(1 \leq j < n_1\).

2. For \(1 \leq j < n_1\) and \(j < s < j + m - n_1 - 1\), rewrite the third dual constraint as

\[
 u_{j,s-1} + w_{js} - u_{js} - w_{j+1,s+1} = 0 
\]

and sum up for \(s = j + 1, \ldots, t\). Adding then the first dual constraint one gets:

\[
 \sum_{h=j+1}^{s+1} x_{j+1,h} - \sum_{h=j}^{s} x_{jh} \leq 0 
\]

i.e., inequalities (3.16).
Projection 2

1. Rewriting the third dual constraints for $j \in A$

   \[ u_{j,s-1} + w_{js} - u_{js} - w_{j+1,s+1} = 0 \]

   and summing them up for $s \in T$, $s \geq j$, we obtain

   \[ \sum_{h=j}^{j+m-n_1} w_{jh} = \sum_{h=j+1}^{j+m-n_1+1} w_{j+1,h} \]

2. Rewriting the fourth dual constraints

   \[ u_{n_1,s-1} + w_{n_1s} - u_{n_1s} = 0 \]

   summing them up for $s \geq n_1$ and adding the result to the sixth dual constraint

   \[ u_{n_1,m-1} + w_{n_1m} = 1 \]

   one gets

   \[ \sum_{h=n_1}^{m} w_{n_1h} = 1 \]

   Putting together the last equation and the one obtained at step 1, we end up with

   \[ \sum_{h=j}^{j+m-n_1} w_{jh} = 1 \]  \hspace{1cm} (3.21)

Given that (3.10) is equal to the objective function of (3.20) when $x = w$, we have proved the following result.

**Theorem 3.3.1.** Inequalities (3.16) and (3.21) plus non-negativity constraints

\[ x_{jt} \geq 0 \ (j \in A, j \neq n_1, j < t < j + m - n_1) \]

and

\[ x_{n_1t} \geq 0 \ (t \geq n_1) \]

completely describe the polyhedron of CSP$_1$.

**Proof.** Constraints of (3.20) constitute a complete description of its integer hull and its projection on the subspace defined by $u = 0$ corresponds to the appropriate constraints. See above. \hfill \Box
Chapter 4

Polyhedral study of one-machine scheduling problem: the case of two users

In this chapter we will suppose that $n = 2$ and separately cover the study of PSP$_2$, SP$_2$ and CSP$_2$. In 4.1, we will give a time-indexed formulation for PSP$_2$, prove that the polyhedron defined as the convex hull of its feasible solutions is full-dimensional, classify several valid inequalities with integral coefficients and show that they are facets of this polytope. In 4.2 we will study SP$_2$ and use the technique of dynamic programming reformulation in order to describe several facets-inducing inequalities of the corresponding polyhedron. Finally, in 4.3 we will extend the facets obtained for SP$_2$ to the convex hull of feasible solutions of problem CSP$_2$.

4.1 PSP$_2$

In this section, we study the polyhedron when two users are in the system. The notation is as follows:

- $J_1 = A$, $x^1_{it} = x_{it}$, $f^1_i(t) = a_i(t)$ for all $i \in A$ and $t \in T$
- $J_2 = B$, $x^2_{jt} = y_{jt}$, $f^2_j(t) = b_j(t)$ for all $j \in B$ and $t \in T$

Recalling Theorem (3.2.1) and Observation (3.2.1), the time-indexed formulation for PSP$_2$ is
\[
\max \sum_{i \in A} \sum_{t \geq i} a_i(t) x_{it} + \sum_{j \in B} \sum_{t \geq j} b_j(t) y_{jt} \quad (4.1)
\]

\[
\sum_{t \geq 1} x_{1t} \leq 1 \quad (4.2)
\]

\[
\sum_{t \geq 1} y_{1t} \leq 1 \quad (4.3)
\]

\[
\min\{n_1, t\} \sum_{i=1}^{t} x_{it} + \min\{n_2, t\} \sum_{j=1}^{t} y_{jt} \leq 1 \quad \forall t \in T \quad (4.4)
\]

\[
\sum_{s=i}^{t} x_{is} - \sum_{s=i-1}^{t-1} x_{i-1,s} \leq 0 \quad \forall i \in A, i \neq 1, t \geq i \quad (4.5)
\]

\[
\sum_{s=j}^{t} y_{js} - \sum_{s=j-1}^{t-1} y_{j-1,s} \leq 0 \quad \forall j \in B, j \neq 1, t \geq j \quad (4.6)
\]

\[
x_{it}, x_{n_1, n_1} \geq 0 \quad \forall i \in A, t \geq i \quad (4.7)
\]

\[
y_{jt}, y_{n_2, n_2} \geq 0 \quad \forall j \in B, t \geq j \quad (4.8)
\]

\[
x_{it} \text{ integer} \quad \forall i \in A, t \geq i \quad (4.9)
\]

\[
y_{jt} \text{ integer} \quad \forall j \in B, t \geq j \quad (4.10)
\]

Let \( S \) be the polyhedron defined by inequalities (4.2)-(4.8) and \( P_S \) the convex hull of all vectors in \( S \) having integer components \( x_{it}, y_{jt} \).

**Theorem 4.1.1.** \( P_S \) is full-dimensional.

**Proof.** Suppose that \( P_S = \{ z \in P_S : \pi^T z = 0 \} \). Let us show that \( \pi = 0 \).

Let \( z = (x, y) \) and denote as \( \pi_{it}^A \) the first \( (m+1)n_1 - \frac{n_1(n_1+1)}{2} \) components of \( \pi \) (corresponding to \( A \)), and as \( \pi_{jt}^B \) the remaining \( (m+1)n_2 - \frac{n_2(n_2+1)}{2} \) components (corresponding to \( B \)). For unit vectors, use the same notation as in Theorem 3.2.4.

For all \( i \in A, i \geq 2 \), define

\[
z_{it}^A = (e_{1t}, 0) \quad \forall t \in T
\]

\[
z_{it}^A = \left( \sum_{h=1}^{i-1} e_{hh} + e_{it}, 0 \right) \quad \forall t \in T, t \geq i
\]

Since the above vectors belong to \( S \) and their components are integer, then \( \pi^T z_{it}^A = 0 \) for \( i \in A \) and \( t \in T, t \geq i \), and it follows

\[
\sum_{h=1}^{i-1} \pi_{ih}^A + \pi_{it}^A = 0
\]
Hence, \( \pi^A_{it} = 0 \), for all \( i \in A \) and \( t \geq i \).

In a similar way, for all \( j \in B \), \( j \geq 2 \), define

\[
\begin{align*}
\mathbf{z}^t_B & = (0, \mathbf{e}^t) \quad \forall t \in T \\
\mathbf{z}^t_B & = (0, \sum_{h=1}^{j-1} \mathbf{e}^{j-1,h-1} + \mathbf{e}^{jt}) \quad \forall t \in T, t \geq j
\end{align*}
\]

From \( \pi^T \mathbf{z}^t_B = 0 \), for all \( j \in B \) and \( t \geq j \) it follows

\[
\sum_{h=1}^{j-1} \pi^B_{hh} + \pi^B_{jt} = 0
\]

Hence, \( \pi^B_{jt} = 0 \) and the theorem is proved.

Observe that the result of Theorem 4.1.1 holds for \( \text{PSP}_n \).

**Theorem 4.1.2. Inequalities**

\[
\begin{align*}
\sum_{s \geq 1} x_{1s} & \leq 1 \\
\sum_{s \geq 1} y_{1s} & \leq 1 \\
x_{it}, x_{n_i} & \geq 0 \quad \forall i \in A, \forall t > i \\
y_{jt}, y_{n_j} & \geq 0 \quad \forall j \in B, \forall t > j
\end{align*}
\]

induce facets of \( P_S \).

**Proof.** See [18].

**Theorem 4.1.3. Inequalities (4.5) and (4.6) induce facets of \( P_S \).**

**Proof.** Similar to Theorem 3.2.4. In fact, one can separately repeat the procedure illustrated in the theorem for both users.

None of the facet-defining inequalities found so far involves variables associated with both \( A \) and \( B \). The first family of inequalities with nonzero coefficients in both sets of variables is given by the following theorem.

**Theorem 4.1.4. Inequalities (4.4) induce facets of \( P_S \).**
Proof. For all \( t \in T \), let

\[
F_t = \{ z = (x, y) \in P_S : \quad \sum_{i=1}^{\min\{n_1, t\}} x_{it} + \sum_{j=1}^{\min\{n_2, t\}} y_{jt} = 1 \}
\]

and let \( \pi^T z = \pi_0 \) be any hyperplane containing \( F_t \). We next show that

\[
\pi^T z = \pi_0 (\sum_{i=1}^{\min\{n_1, t\}} x_{it} + \sum_{j=1}^{\min\{n_2, t\}} y_{jt})
\]

Define

\[
z_A^{1t} = (e_1^t, 0) \quad z_B^{1t} = (0, e_1^t)
\]

Since these vectors belong to \( F_t \) and have integer components, we solve \( \pi^T z_A^{1t} = \pi_0 \) and \( \pi^T z_B^{1t} = \pi_0 \) and, for all \( t \) obtain

\[
\pi^A_{1t} = \pi^B_{1t} = \pi_0
\]

For all \( i = 1, \ldots, \min\{n_1, t\} \), \( s \geq i \), \( s \neq t \) and \( j = 1, \ldots, \min\{n_2, t\} \), \( k \geq j \), \( k \neq t \), define

\[
z_A^{is} = \left( \sum_{h=1}^{i-1} e^{hh} + e^{is}, 0 \right) + (0, e_1^t) \quad z_B^{jk} = (e_1^t, 0) + (0, \sum_{h=1}^{j-1} e^{hh} + e^{jk})
\]

again, plugging \( z_A^{is}, z_B^{jk} \) into \( \pi^T z = \pi_0 \) and solving, one gets

\[
\pi^A_{is} = \pi^B_{jk} = 0
\]

For all \( i = 2, \ldots, \min\{n_1, t\} \) and \( j = 2, \ldots, \min\{n_2, t\} \), take now \( z_A^{it} \) and \( z_B^{jt} \) as in Theorem 4.1.1. Then

\[
\pi^A_{it} = \pi^B_{jt} = \pi_0
\]

Finally, if \( \min\{n_1, t\} = t \) \( (\min\{n_2, t\} = t) \), then solve \( \pi^T z_A^{is} = \pi_0 \) \( (\pi^T z_B^{js} = \pi_0) \) for all \( i > t \) \( (j > t) \) and \( s \geq i \) \( (s \geq j) \). Hence

\[
\pi^A_{is} = 0 \quad (\pi^B_{js} = 0)
\]

\( \square \)
Now, we derive next facet-inducing inequalities of $P_S$. The following inequalities have the expression

$$x(S_1) + y(S_2) \leq 1 \quad (4.11)$$

where $S_k \subseteq J_k \times T$. That is, in a feasible schedule at most one variable of $S_1 \cup S_2$ can be set to 1. Subsets $S_k$ are chosen in such a way that if inequality (4.11) is violated by a solution, then so are the precedence constraints.

**Theorem 4.1.5. Inequalities**

$$\sum_{s=i}^{t} x_{is} + \sum_{s=t-i+1}^{t} y_{t-i+1,s} \leq 1 \quad (4.12)$$

for $i \in A$, $i \geq 2$ and $t \in T$, $t \geq i$, and

$$\sum_{s=j}^{t} y_{js} + \sum_{s=t-j+1}^{t} x_{t-j+1,s} \leq 1 \quad (4.13)$$

for $j \in B$, $j \geq 2$ and $t \in T$, $t \geq j$, induce facets of $P_S$.

**Proof.** For $i \in A$, $i \geq 2$ and $t \in T$, $t \geq i$, let

$$F_{it} = \{z = (x, y) \in P_S : \sum_{s=i}^{t} x_{is} + \sum_{s=t-i+1}^{t} y_{t-i+1,s} = 1\}$$

and let $\pi^T z = \pi_0$ be any hyperplane containing $F_{it}$. We next show that

$$\pi^T z = \pi_0 [\sum_{s=i}^{t} x_{is} + \sum_{s=t-i+1}^{t} y_{t-i+1,s}]$$

For $i \leq s \leq t$ and $t - i + 1 \leq h \leq t$, take the vectors $z^A_{is}$ and $z^B_{t-i+1,h}$ as in Theorem (4.1.1). Since they belong to $F_{it}$ and their components are integer, we solve $\pi^T z^A_A = \pi_0$ and $\pi^T z^B_{t-i+1,h} = \pi_0$ and obtain

$$\pi^A_{is} = \pi_0 - \sum_{h=1}^{i-1} \pi^A_{hh} \quad i \leq s \leq t$$

$$\pi^B_{t-i+1,h} = \pi_0 - \sum_{h=1}^{i-1} \pi^B_{hh} \quad t - i + 1 \leq h \leq t$$

For $2 \leq s \leq t + 1 - i$, define

$$z^A_{is} = \left( \sum_{h=0}^{i-1} e^{1+h,s+h}, 0 \right)$$
and again, for $2 \leq k \leq i - 1$ and $k < s \leq t + k - i$

$$z_A^{ks} = \left( \sum_{h=1}^{k-1} e^{kh} + \sum_{h=0}^{i-k} e^{k+h,s+h}, 0 \right)$$

Since $k + 1 \leq s \leq t + k - i$, then $i + 1 \leq s + i - k \leq t$ and all the above vectors belong to $F_{it}$. Solving the associated systems, for $1 \leq k \leq i - 1$ it follows

$$\pi_{kk}^A = \pi_{k,k+1}^A = \ldots = \pi_{k,t+k-1}^A$$

In similar way, for $2 \leq s \leq i$ define

$$z_B^{1s} = (0, \sum_{h=0}^{t-i} e^{1+h,s+h})$$

and, for $2 \leq j \leq t - i$ and $j < s \leq i + j - 1$

$$z_B^{js} = (0, \sum_{h=1}^{j-1} e^{kh} + \sum_{h=0}^{t-i-j+1} e^{i+h,s+h})$$

Solving the systems, for $1 \leq j \leq t - i$ it follows

$$\pi_{jj}^B = \pi_{j,j+1}^B = \ldots = \pi_{j,t+j-1}^B$$

For $1 \leq k \leq i - 1$ and $1 \leq j \leq t - i$, define

$$z_A^{kk} = \left( \sum_{h=1}^{k} e^{kh}, 0 \right) + \left( 0, \sum_{h=1}^{t-i+1} e^{h,k+h} \right)$$

$$z_B^{jj} = \left( 0, \sum_{h=1}^{j} e^{kh} \right) + \left( \sum_{h=1}^{i} e^{h,j+h}, 0 \right)$$

and, plugging $z_A^{kk}, z_B^{jj}$ into $\pi^T z = \pi_0$, one gets

$$\pi_{kk}^A = 0 \quad \pi_{jj}^B = 0$$

for $k \in A$, $k \leq i - 1$, and $j \in B$, $j \leq t - i$. Exploiting these values of $\pi$, one rewrites

$$\pi_{ks}^A = 0 \quad k \leq s \leq t - i + k$$

$$\pi_{js}^B = 0 \quad j \leq s \leq i + j - 1$$

for $k \in A$, $k \leq i - 1$, and $j \in B$, $j \leq t - i$. 


For \( t - i + 2 \leq s \leq m \), define
\[
\mathbf{z}_{is}^A = (0, \sum_{h=1}^{t-i+1} e^{hh}) + (\mathbf{e}^1 s, 0)
\]
and again, for \( 2 \leq k \leq i - 1 \) and \( t + k - i + 1 \leq s \leq m \)
\[
\mathbf{z}_{ks}^A = (0, \sum_{h=1}^{t-i+1} e^{hh}) + (\sum_{h=1}^{k-1} e^{h,t-i+1+h} + \mathbf{e}^k s, 0)
\]
Solving \( \pi^T \mathbf{z}_{ks}^A = \pi_0 \), one obtains
\[
\pi_{ks}^A = 0
\]
for \( k \in A, k \leq i - 1 \) and \( t + k - i + 1 \leq s \leq m \).

In similar way, for \( i + 1 \leq s \leq m \) define
\[
\mathbf{z}_{is}^B = (\sum_{h=1}^{i} e^{hh}, 0) + (0, \mathbf{e}^1 s)
\]
and again, for \( 2 \leq j \leq t - i \) and \( i + j \leq s \leq m \)
\[
\mathbf{z}_{js}^B = (\sum_{h=1}^{i} e^{hh}, 0) + (0, \sum_{h=1}^{j-1} e^{h,i+h} + \mathbf{e}^j s)
\]
Solving the associated systems, one gets
\[
\pi_{js}^B = 0
\]
for \( j \in B, j \leq t - i \) and \( i + j + 1 \leq s \leq m \).

For \( t + 1 \leq s \leq m \), define
\[
\mathbf{z}_{is}^A = (0, \sum_{h=1}^{t-i+1} e^{hh}) + (\sum_{h=1}^{i-1} e^{h,t-i+h+1} + \mathbf{e}^i s, 0)
\]
\[
\mathbf{z}_{t-i+1,s}^B = (\sum_{h=1}^{i} e^{hh}, 0) + (0, \sum_{h=1}^{t-i} e^{h,i+h} + \mathbf{e}^{t-i+1,s})
\]
Solving the associated systems, it follows that all the components of \( \pi \) corresponding to the above vectors equal 0.

Finally, for \( k \in A, k \geq i + 1 \) and \( s \in T, s \geq k \) define
\[
\mathbf{z}_{ks}^A = (\sum_{h=1}^{k-1} e^{hh} + \mathbf{e}^k s, 0)
\]
and obtain
\[
\pi_{ks}^A = 0
\]
In similar way, for \( j \in B, j \geq t - i + 2 \) and \( s \in T, s \geq j \) define
\[
z_{js}^B = (0, \sum_{h=1}^{j-1} e^{hh} + e^{js})
\]
and obtain
\[
\pi_{js}^B = 0
\]
From the values of \( \pi_{it}^A \) it and \( \pi_{jt}^B \) the result follows. In similar way, one obtains inequalities (4.13).

The support graph of inequality (4.12) is depicted in Figure 4.1. First observe that at most one variable corresponding to a “left” (“right”) edge can take value 1 because the corresponding job cannot be assigned more than once. Now, if two variables, say \( x_is \) and \( y_{t-i+1,s'} \) (\( y_{js} \) and \( x_{t-j+1,s'} \)), i.e. corresponding to one left and one right edge, take value 1, the number of time slots before \( \max\{s, s'\} \) is not large enough to schedule all the jobs preceding jobs \( i (j) \) of user \( A (B) \) and job \( t - i + 1 (t - j + 1) \) of user \( B (A) \).

This interpretation holds for all the inequalities that we are going to present.

**Theorem 4.1.6.** Inequalities

\[
x_{it} + \sum_{s=i+1}^{t} x_{i+1,s} + y_{t-i,t} + \sum_{s=t-i+1}^{t} y_{t-i+1,s} \leq 1 \tag{4.14}
\]

for \( i \in A \) and \( t \in T, t \geq i + 2 \), and

\[
y_{jt} + \sum_{s=j+1}^{t} y_{j+1,s} + x_{t-j,t} + \sum_{s=t-j+1}^{t} x_{t-j+1,s} \leq 1 \tag{4.15}
\]

for \( j \in B \) and \( t \in T, t \geq j + 2 \), induce facets of \( P_S \).

**Proof.** For \( i \in A \) and \( t \in T, t \geq i + 2 \), let

\[
F_{it} = \{z = (x, y) \in P_S : x_{it} + \sum_{s=i+1}^{t} x_{i+1,s} + y_{t-i,t} + \sum_{s=t-i+1}^{t} y_{t-i+1,s} = 1\}
\]

and let \( \pi^Tz = \pi_0 \) be any hyperplane containing \( F_{it} \). We next show that

\[
\pi^Tz = \pi_0[x_{it} + \sum_{s=i+1}^{t} x_{i+1,s} + y_{t-i,t} + \sum_{s=t-i+1}^{t} y_{t-i+1,s}]
\]
For $i + 1 \leq s \leq t$, define

\[
z_{it}^A = (\sum_{h=1}^{i} e^{h,t-i+h}, 0) \quad z_{i+1,s}^A = (\sum_{h=1}^{i+1} e^{h,s-i+h-1}, 0)
\]

and, for $t - i + 1 \leq s \leq t$

\[
z_{t-i,t}^B = (0, \sum_{h=1}^{t-i} e^{h,t-i+h}) \quad z_{t-i+1,s}^B = (0, \sum_{h=1}^{t-i+1} e^{h,s-t+i+h-1})
\]

Solving the associated systems, for $i + 1 \leq s \leq t$ one gets

\[
\pi_{it}^A = \pi_0 - \sum_{h=1}^{i-1} \pi_{h,t-i+h}^A \quad \pi_{i+1,s}^A = \pi_0 - \sum_{h=1}^{i} \pi_{h,s-i+h-1}^A
\]
and, for \( t - i + 1 \leq s \leq t \)

\[
\pi^B_{t-i,s} = \pi^B_{t-i-1} - \sum_{h=1}^{t-i-1} \pi^B_{t-i-1,h} + \pi^B_{t-i+1,s} - \sum_{h=1}^{t-i} \pi^B_{t-i+1,s-t+i+h-1}
\]

For \( s \in T \) and \( k \in A, k \geq i + 3 \), define

\[
\begin{align*}
\mathbf{z}^{i+1,s}_{t-i} & = \mathbf{z}^{i+1,s}_{t-i} + (\mathbf{e}^{i+1,s}, \mathbf{0}) & s \geq t + 1 \\
\mathbf{z}^{i+2,s}_{t-i} & = \mathbf{z}^{i+2,s}_{t-i} + (\mathbf{e}^{i+2,s}, \mathbf{0}) & s \geq i + 2 \\
\mathbf{z}^{ks}_{t-i} & = \mathbf{z}^{i+1,i+1}_{t-i} + (\sum_{h=i+2}^{k-1} \mathbf{e}^{hh} + \mathbf{e}^{ks}, \mathbf{0}) & s \geq k
\end{align*}
\]

and, for \( k \in B, 2 \leq k \leq t - i - 1 \),

\[
\begin{align*}
\mathbf{z}^{1s}_{t-i} & = \mathbf{z}^{1s}_{t-i} + (\mathbf{0}, \mathbf{e}^{1s}) & s \leq i \\
\mathbf{z}^{ks}_{t-i} & = (\sum_{h=1}^{k-1} \mathbf{e}^{hh} + \mathbf{e}^{ks}, \mathbf{0}) + \mathbf{z}^{i+1,i+1}_{t-i} & k \leq s \leq i \\
\mathbf{z}^{it}_{t-i} & = (\sum_{h=1}^{i} \mathbf{e}^{hh}, \mathbf{0}) + \mathbf{z}^{i+1,i+1}_{t-i} &
\end{align*}
\]

and again, for \( k \in B, 2 \leq k \leq t - i - 1 \)

\[
\begin{align*}
\mathbf{z}^{1s}_{t-i} & = (\mathbf{0}, \mathbf{e}^{1s}) + \mathbf{z}^{i+1,i+1}_{t-i} & s \leq t - i \\
\mathbf{z}^{ks}_{t-i} & = (\sum_{h=1}^{k-1} \mathbf{e}^{hh} + \mathbf{e}^{ks}, \mathbf{0}) + \mathbf{z}^{i+1,i+1}_{t-i} & k \leq s \leq t - i \\
\mathbf{z}^{it}_{t-i,t-i} & = (\sum_{h=1}^{t-i} \mathbf{e}^{hh}) + \mathbf{z}^{i+1,i+1}_{t-i} & s \leq t - i \\
\end{align*}
\]

Solving the associated systems, it follows that all the components in \( \pi \) corresponding to the above vectors equal 0.

For \( s \in T \) and \( k \in A, 2 \leq k \leq i - 1 \), define

\[
\begin{align*}
\mathbf{z}^{ks}_{t-i} & = (\sum_{h=1}^{k-1} \mathbf{e}^{hh} + \mathbf{e}^{ks}, \mathbf{0}) + \mathbf{z}^{i+1,i+1}_{t-i} & k \leq s \leq i \\
\mathbf{z}^{it}_{t-i} & = (\sum_{h=1}^{i} \mathbf{e}^{hh}, \mathbf{0}) + \mathbf{z}^{i+1,i+1}_{t-i} &
\end{align*}
\]

and again, for \( k \in B, 2 \leq k \leq t - i - 1 \)

\[
\begin{align*}
\mathbf{z}^{ks}_{t-i} & = (\sum_{h=1}^{k-1} \mathbf{e}^{hh} + \mathbf{e}^{ks}, \mathbf{0}) + \mathbf{z}^{i+1,i+1}_{t-i} & k \leq s \leq t - i \\
\mathbf{z}^{it}_{t-i,t-i} & = (\sum_{h=1}^{t-i} \mathbf{e}^{hh}) + \mathbf{z}^{i+1,i+1}_{t-i} & s \leq t - i \\
\end{align*}
\]

Solving the associated system, it follows that all the components in \( \pi \) corresponding to the above vectors equal 0.
For $s \in T$, define again

$$z_{is}^A = \left( \sum_{h=1}^{i+1} e_{h,s-i+h}, 0 \right) \quad i+1 \leq s \leq t-1$$

$$z_{is}^A = \left( \sum_{h=1}^{i-1} e_{hb} + e_{is}, 0 \right) + z_{B}^{t-i,t} \quad s \geq t+1$$

$$z_{B}^{t-i,s} = \left( 0, \sum_{h=1}^{t-i+1} e_{h,s-t+i+h} \right) \quad t-i+1 \leq s \leq t-1$$

$$z_{B}^{t-i,s} = z_{A}^{it} + \left( 0, \sum_{h=1}^{t-i-1} e_{hh} + e^{t-i,s} \right) \quad s \geq t+1$$

Also in this case, all the components in $\pi$ corresponding to the above vectors equal 0.

For $s \in T$ and $k \in A$, $2 \leq k \leq i-1$ define

$$z_{is}^A = \left( e_{is}, 0 \right) + \left( 0, \sum_{h=1}^{i} e_{hb} + \sum_{h=1}^{t-2i+1} e_{i+h,s+h} \right)$$

$$i+1 \leq s \leq 2i-1$$

$$z_{is}^A = \left( \sum_{h=1}^{k-1} e_{h,i-k+h+1} + e_{ks}, 0 \right) + \left( 0, \sum_{h=1}^{i-k+1} e_{hh} + \sum_{h=2}^{t-2i+k+1} e_{t-k+h,s+h-1} \right)$$

$$i+1 \leq s \leq 2i-k$$

and again, for $k \in B$, $2 \leq k \leq t-i-1$

$$z_{is}^A = \left( 0, e_{is} \right) + \left( \sum_{h=1}^{t-i} e_{bh} + \sum_{h=1}^{2i-t+1} e^{t-i+h,s+h} + e_{is}, 0 \right)$$

$$t-i+1 \leq s \leq 2t-2i-1$$

$$z_{is}^A = \left( 0, \sum_{h=1}^{k-1} e_{h,t-i-k+h+1} + e_{ks} \right) + \left( \sum_{h=1}^{t-i-k+1} e_{hh} + \sum_{h=2}^{2i-t+k+1} e^{t-i-k+h,s+h-1}, 0 \right)$$

$$t-i+1 \leq s \leq 2t-2i-k$$

Solving the associated system, it follows that all the components in $\pi$ corresponding to the above vectors equal 0.

For $s \in T$ and $k \in A$, $2 \leq k \leq i-1$ define

$$z_{is}^A = e_{is} + \sum_{h=1}^{s-1} f_{hh} + \sum_{h=s}^{t-i+1} f_{h,h+1}$$
4 Polyhedral Study: the case of two users

\[ z^A_s = \sum_{h=1}^{k-1} e^{h,s-k+h} + e^{k,s} + \sum_{h=1}^{s-k} t^{h} + \sum_{h=s-k+1}^{t-i+1} f^{h,h+k} \]

\[ 2i - k + 1 \leq s \leq t - i + k \]

and again, for \( 2 \leq k \leq i - 1 \)

\[ z^B_s = e^{1s} + z^{l-i+1,t-i+1}_B \]

\[ s \geq t - i + 2 \]

\[ z^A_s = \sum_{h=1}^{k-1} e^{h,t-i+h+1} + e^{ks} + z^{l-i+1,t-i+1}_B \]

\[ s \geq t - i + k + 1 \]

In symilar way, for \( k \in B, 2 \leq k \leq t - i - 1 \) define

\[ z^B_s = f^{1s} + \sum_{h=1}^{s-1} e^{hh} + \sum_{h=s}^{i+1} e^{h,h+1} \]

\[ 2t - 2i \leq s \leq i + 1 \]

\[ z^B_s = \sum_{h=1}^{k-1} f^{h,s-k+h} + f^{ks} + \sum_{h=1}^{s-k} e^{hh} + \sum_{h=s-k+1}^{i+1} e^{h,h+k} \]

\[ 2t - 2i - k + 1 \leq s \leq i + k \]

and again, for \( 2 \leq k \leq t - i - 1 \)

\[ z^B_s = f^{1s} + z^{i+1,i+1}_B \]

\[ s \geq i + 2 \]

\[ z^B_s = \sum_{h=1}^{k-1} f^{h,i+h+1} + f^{ks} + z^{i+1,i+1}_A \]

\[ s \geq i + k + 1 \]

Solving the associated system, one gets that all the components in \( \pi \) corresponding to the above vectors equal 0.

From the components of \( \pi \) just computed, one writes

\[ \pi^A_{it} = \pi^A_{k+1,s} = \pi_0 \quad \pi^B_{t-k,t} = \pi^B_{t-k+1,h} = \pi_0 \]

for \( s, h \in T, i + 1 \leq s \leq t \) and \( t - i + 1 \leq h \leq t \), and the theorem is proved.

For inequalities (4.15), the proof is analogous. \( \square \)
Here again at most one $x$-variable ($y$-variable) appearing in (4.14) or (4.15) can take value 1. Further if one $x$-variable and one $y$-variable take value 1, then there are not enough time slots to schedule the previous jobs. The support graph of inequality (4.14) is depicted in Figure 4.2.

**Theorem 4.1.7.** Inequalities

\[
\min\{n_1, t\} \sum_{j=i} \sum_{t-j} x_{jt} + \sum_{j=1}^{t-i} y_{jt} + \sum_{s=t-i+1}^{t} y_{t-i+1, s} \leq 1 \quad (4.16)
\]

for $i \in A$, $i \geq 2$ and $t \in T$, $1 \leq t - i \leq n_2 - 1$, and

\[
\min\{n_2, t\} \sum_{i=j} \sum_{t-j} y_{it} + \sum_{i=1}^{t-j} x_{it} + \sum_{s=t-j+1}^{t} x_{t-j+1, s} \leq 1 \quad (4.17)
\]

for $j \in B$, $j \geq 2$ and $t \in T$, $1 \leq t - j \leq n_1 - 1$, induce facets of $P_S$.
Proof. For $i \in A$, $i \geq 2$ and $t \in T$, $t > i$, let

$$F_{it} = \{ z = (x, y) \in P_S : \sum_{j=i}^{t} x_{jt} + \sum_{j=1}^{t-i} y_{jt} + \sum_{s=t-i+1}^{t} y_{t-i+1,s} = 1 \}$$

and let $\pi^T z = \pi_0$ be any hyperplane containing $F_{it}$. We next show that

$$\pi^T z = \pi_0 \left[ \sum_{j=i}^{t} x_{jt} + \sum_{j=1}^{t-i} y_{jt} + \sum_{s=t-i+1}^{t} y_{t-i+1,s} \right]$$

Suppose that $\min \{ n_1, t \} = n_1$ (if it is not, the procedure is analogous). Define the vector

$$z_{1t}^B = (0, e_1^1)$$

belonging to $F_{it}$ and having integer components. Solving the system $\pi^T z_{1t}^B = \pi_0$, one obtains

$$\pi_{1t}^B = \pi_0$$

For $k \in A$ and $s \in T$, $s \geq k$ and $s \neq t$, define

$$z_{ks}^A = \left( \sum_{h=1}^{k-1} e^{h,s} + e^{k,s}, 0 \right) + (0, e_1^t)$$

and solve $\pi^T z_{ks}^A = \pi_0$. It follows

$$\pi_{ks}^A = 0$$

For $k \in A$, $i \leq k \leq \min \{ n_1, t \} = n_1$, define

$$z_{kt}^A = \left( \sum_{h=1}^{k-1} e^{h,t} + e^{k,t}, 0 \right)$$

and solve $\pi^T z_{kt}^A = \pi_0$. It follows

$$\pi_{kt}^A = \pi_0$$

For $j \in B$, $j \leq t - i$ and $s \in T$, $j + 1 \leq s \leq t - 1$, define

$$z_{jj}^B = \left( \sum_{h=1}^{t-j} e^{h,j+h} + \sum_{h=1}^{j} e^{h} \right)$$

$$z_{js}^B = \left( \sum_{h=1}^{s-j} e^{h} + \sum_{h=1}^{t-s} e^{s-j+h,s+h}, 0 \right) + (0, \sum_{h=1}^{j} e^{h,s-j+h})$$
Solving $\pi^T z_{js}^B = \pi_0$, it follows

$$\pi_{js}^B = 0 \quad j \leq s \leq t - 1$$

For $j \in B, 2 \leq j \leq t - i + 1$, define

$$z_{jt}^B = (0, \sum_{h=1}^{j-1} e^{hh} + e^{jt})$$

and, for $s \in T, t - i + 1 \leq s \leq t - 1$

$$z_{t-i+1,s}^B = (0, \sum_{h=1}^{t-i} e^{hh} + e^{t-i+1,s})$$

Plugging $z_{jt}^B, z_{t-i+1,s}^B$ into $\pi^T z = \pi_0$, one gets

$$\pi_{jt}^B = \pi_0 \quad \pi_{t-i+1,s}^B = \pi_0$$

For $k \in A, 1 < k \leq i - 1$, define

$$z_{1t}^A = (e^{1t}, 0) + (0, \sum_{h=1}^{t-i+1} e^{hh})$$

$$z_{kt}^A = (\sum_{h=1}^{k} e^{h,t-k+h}, 0) + (0, \sum_{h=1}^{t-i+1} e^{hh})$$

Solving $\pi^T z_{1t}^A = \pi_0$, it follows

$$\pi_{kt}^A = 0 \quad k \leq i - 1, k \neq 1$$

For $j \in B, 1 < j \leq t - i + 1$ and $s \in T, s \geq t + 1$, define

$$z_{1s}^B = (\sum_{h=1}^{t} e^{hh}, 0) + (0, e^{1s})$$

$$z_{js}^B = (0, \sum_{h=1}^{j-1} e^{h,t-j+h+1} + e^{js})$$

Solving $\pi^T z_{1s}^B = \pi_0$, it follows

$$\pi_{js}^B = 0$$

Finally, for $j \in B, j \geq t - i + 2$ and $s \in T, s \geq j$, defining

$$z_{js}^B = (0, \sum_{h=1}^{j-1} e^{hh} + e^{js})$$
and solving $\pi^T z^j_s = \pi_0$, one obtains

$$\pi^B_j s = 0$$

and the theorem is proved. For inequalities (4.17), the proof is analogous.  

The interpretation to these inequalities is similar to that of the previous ones.

A further family of facets contains inequalities of the form

$$-x_{it} + x(S_1) + y(S_2) \leq 1 \quad \text{or} \quad -y_{jt} + x(S_1) + y(S_2) \leq 1 \quad (4.18)$$

for suitable values of $t \in T$ and any $i \in A$, $j \in B$. The meaning of inequalities (4.18) is the same as (4.11), but now the number of 1’s in $S_1 \cup S_2$ depends on $x_{it}$ or $y_{jt}$.

**Theorem 4.1.8.** Inequalities

$$-y_{jt} + \sum_{h=t-j}^{t-1} x_{t-j,h} + x_{t-j,s} + y_{s-t+j,s} + \sum_{h=s-t+j+1}^{s} y_{s-t+j+1,h} \leq 1 \quad (4.19)$$

for $j \in B$, $j \neq n_2$, $s, t \in T$, $t \geq j + 2$ and $s > t$, and

$$-x_{it} + \sum_{h=t-i}^{t-1} y_{t-i,h} + y_{t-i,s} + x_{s-t+i,s} + \sum_{h=s-t+i+1}^{s} x_{s-t+i+1,h} \leq 1 \quad (4.20)$$

for $i \in A$, $i \neq n_1$, $t \in T$, $t \geq i + 2$ and $s > t$, induce facets of $P_S$.

**Proof.** For more semplicity, the theorem is proved for $j = 1$. For any $j \in B$, the proof is analogous.

For $s, t \in T$, $t \geq 3$ and $s > t$, let

$$F^s_{1t} = \{z = (x, y) \in P_S : -y_{1t} + x_{t-1,t-1} + x_{t-1,s} + y_{s-t+1,s} + \sum_{h=s-t+2}^{s} y_{s-t+2,h} = 1\}$$

and let $\pi^T z = \pi_0$ be any hyperplane containing $F^s_{1t}$. We next show that

$$\pi^T z = \pi_0(-y_{1t} + x_{t-1,t-1} + x_{t-1,s} + y_{s-t+1,s} + \sum_{h=s-t+2}^{s} y_{s-t+2,h}).$$

In the rest of the proof, assume that $s < 2t - 3$ (if it is not, the theorem can be proved in a similar way).
Defining
\[ z_B^{1t} = (\sum_{h=1}^{t-1} e^{hh}, 0) + (0, \sum_{h=1}^{s-t+1} e^{h,t+h-1}) \]
and solving \( \pi^T z_B^{1t} = \pi_0 \), one obtains
\[ \pi_B^{1t} = \pi_0 - \sum_{h=1}^{t-1} \pi_A^{hh} - \sum_{h=2}^{s-t+1} \pi_B^{h,t+h-1} \]

For \( j \in B, j \geq 2 \), and \( k \in T \) define
\[ z_B^{1k} = (\sum_{h=1}^{t-1} e^{hh}, 0) + (0, e^{1k}) \quad k \geq t+1 \]
\[ z_B^{jk} = (\sum_{h=1}^{t-1} e^{hh}, 0) + (0, \sum_{h=1}^{s-t+j} e^{h,k-j+h}) \quad k \geq t+j \]

Solving the associated systems, it follows
\[ \pi_B^{1k} = \pi_0 - \sum_{h=1}^{t-1} \pi_A^{hh} \quad k \geq t+1 \quad (4.21) \]
\[ \pi_B^{jk} = 0 \quad k \geq t+j \]

For \( k \in T, s-t+2 \leq k \leq s \), defining
\[ z_B^{s-t+1,s} = (0, \sum_{h=1}^{s-t} e^{hh} + e^{s-t+1,s}) \]
\[ z_B^{s-t+2,k} = (0, \sum_{h=1}^{s-t+1} e^{hh} + e^{s-t+2,k}) \]

and solving the associated systems, one gets
\[ \pi_B^{s-t+1,s} = \pi_0 - \sum_{j=1}^{s-t} \pi_B^{jj} \quad \pi_B^{s-t+2,k} = \pi_0 - \sum_{h=1}^{s-t+1} \pi_B^{hh} \]

For \( j \in B, j \geq s-t+3 \) and \( k \in T, j \leq k \leq t+j-1 \) define
\[ z_B^{s-t+2,s+1} = (0, \sum_{h=1}^{s-t} e^{hh} + e^{s-t+1,s} + e^{s-t+2,s+1}) \]
\[ z_B^{jk} = (0, \sum_{h=1}^{j-1} e^{hh} + e^{jk}) \]
and solve the associated systems. It follows

\[ \pi_{s-t+2,s+1}^B = \pi_{jk}^B = 0 \]

For \( j \in B, \, 2 \leq j \leq s - t + 1 \), and \( k \in T, \, j \leq k \leq s - t + 1 \) define

\[ z_{B}^{jk} = (\sum_{h=1}^{t-1} e^{h,s-t+h+1}, 0) + (0, e^{jk}) \]

Plugging the above vectors into \( \pi^T z = \pi_0 \), it follows

\[ \pi_{1k}^B = \pi_0 - \sum_{h=1}^{k-1} \pi_{h,h}^A - \sum_{h=k}^{t-1} \pi_{h,s-t+h+1}^A \quad k \geq 1 \]  

(4.22)

\[ \pi_{jk}^B = 0 \]

Exploting the components of \( \pi \) just obtained and substituting them in the expression of \( \pi_{s-t+1,s}^B \) and \( \pi_{s-t+1,h}^B \), for \( s - t + 2 \leq h \leq s \), one gets

\[ \pi_{s-t+1,s}^B = \pi_{s-t+2,h}^B = \pi_0 - \pi_{11}^B \]  

(4.23)

For \( j \in B, \, j \leq s - t + 1 \), and \( k \in T \), define

\[ z_{B}^{jk} = (\sum_{h=1}^{k-j} e^{hh} + \sum_{h=k-j+1}^{t-1} e^{h,s-t+h+1}, 0) + (0, \sum_{h=1}^{j} e^{h,k-j+h}) \]

\[ \pi_{s-t+1,s}^B = \pi_{s-t+2,h}^B = \pi_0 - \pi_{11}^B \]  

(4.23)

For \( j \in B, \, j \leq s - t + 1 \), and \( k \in T \), define

\[ z_{B}^{jk} = (0, \sum_{h=1}^{j-1} e^{hh} + e^{jk} + \sum_{h=j+1}^{s-t+2} e^{h,t+h-2}) \]

\[ j \neq 1, \, s - t + 2 \leq k \leq s - t + j \]

and again, for \( 2 \leq j \leq s - t \)

\[ z_{B}^{j,t+j-1} = (\sum_{h=1}^{t-2} e^{h,h+1} + e^{t-1,s}, 0) + (0, e^{11} + \sum_{h=2}^{j} e^{h,t+h-1}) \]

Solving the associated systems, one obtains that the only components of \( \pi \) different from 0 are

\[ \pi_{1k}^B = \pi_0 - \sum_{h=1}^{k-1} \pi_{h,h}^A - \sum_{h=k}^{t-1} \pi_{h,s-t+h+1}^A \]  

(4.24)

\[ \pi_{2,t+1}^B = \pi_0 - \pi_{11}^B - \sum_{h=1}^{t-2} \pi_{h,h+1}^A + \pi_{t-1,s}^A \]  

(4.25)
for \( s - t + 2 \leq k \leq t - 1 \). Substituting them in the expression of \( \pi^B_{1t} \), it follows

\[
\pi^B_{1t} = \pi_0 - \sum_{h=1}^{t-1} \pi^A_{hh} - \pi^B_{2,t+1} - \pi^B_{s-t+1,s}
\] (4.26)

One concludes that, thus far, the components of \( \pi \) whose value is different from 0 are represented by (4.21)-(4.26).

Now, define the vectors

\[
z^{t-1,t-1}_A = \left( \sum_{h=1}^{t-1} e^{hh}, 0 \right)
\]

\[
z^{t-1,s}_A = \left( \sum_{h=1}^{t-2} e^{hh} + e^{t-1,s}, 0 \right)
\]

and solve the associated systems. It follows

\[
\pi^A_{t-1,t-1} = \pi^A_{t-1,s} = \pi_0 - \sum_{h=1}^{t-2} \pi^A_{hh}
\]

For \( j \in A, j \geq t, \) and \( k \in T, k \geq j \), consider

\[
z^{jk}_A = \left( \sum_{h=1}^{j-1} e^{hh} + e^{jk}, 0 \right)
\]

and solve the system \( \pi^T z^{jk}_A = \pi_0 \). One gets

\[
\pi^A_{jk} = 0
\]

For \( j \in A, 2 \leq j \leq t - 1, \) and \( k \in T, k \geq s - t + j + 2 \), define

\[
z^{1k}_A = (e^{1k}, 0) + \left( 0, \sum_{h=1}^{s-t+2} e^{hh} \right) \quad k \geq s - t + 3
\]

\[
z^{jk}_A = \left( \sum_{h=1}^{j} e^{h,k-j+h}, 0 \right) + \left( 0, \sum_{h=1}^{s-t+2} e^{hh} \right)
\]

Solving the associated systems and exploiting the components of \( \pi \) previously obtained, it follows

\[
\pi^A_{jk} = 0 \quad j \leq t - 1, \ k \geq s - t + j + 2
\]
For \( j \in A, \ 2 \leq j \leq 2t - s - 3 \), and \( k \in T, \ j \leq k \leq s - t + j + 1 \), define again
\[
\begin{align*}
z_{A}^{1k} &= (e_{1k}^{1},0) + (0, \sum_{h=1}^{s-t+2} e_{h,t+h-2}^{1}) \\
z_{A}^{jk} &= (\sum_{h=1}^{j} e_{h,k-j+h}^{h} , 0) + (0, \sum_{h=1}^{s-t+2} e_{h,t+h-2}^{h})
\end{align*}
\]
and solve the systems obtained plugging the above vectors into \( \pi^{T}z = \pi_{0} \). Exploiting also the values of the others components of \( \pi \), it follows
\[
\begin{align*}
\pi_{1k}^{A} &= \pi_{0} - \pi_{1,t-1}^{B} - \pi_{s-t+2,s}^{B} \quad k \leq s - t + 2 \quad (4.27) \\
\pi_{jk}^{A} &= 0
\end{align*}
\]
Substituting the above components in the expressions of \( \pi_{t-1,t-1}^{A} \) and \( \pi_{t-1,s}^{A} \), it follows
\[
\pi_{t-1,t-1}^{A} = \pi_{t-1,s}^{A} = \pi_{0} - \sum_{h=2t-s-2}^{t-2} \pi_{hh}^{A}
\]
Finally, for \( j \in A, \ 2t - s - 2 \leq j \leq t - 2 \), and \( k \in T, \ k \leq t - 2 \), define
\[
\begin{align*}
\begin{align*}
z_{A}^{jk} &= (\sum_{h=1}^{j} e_{h,k-j+h}^{h} , 0) + (0, \sum_{h=1}^{s-t+2} e_{h,t+h-2}^{h}) \\
z_{A}^{2t-s-2,t-1} &= (\sum_{h=1}^{2t-s-3} e_{h}^{h} + e_{2t-s-2,t-1}^{2}, 0) + (0, \sum_{h=1}^{s-t} e_{h,2t-s+h-2}^{h} + e_{s-t+1,s}^{s})
\end{align*}
\end{align*}
\]
and, for \( j \in A, \ 2t - s - 1 \leq j \leq t - 2 \)
\[
\begin{align*}
\begin{align*}
z_{A}^{jk} &= (\sum_{h=1}^{t-k+j-2} e_{h}^{h} + \sum_{h=t-k+j-1}^{j} e_{h,h+k-j}^{h}, 0) + \\
&\quad (0, \sum_{h=1}^{k-j} e_{h,j-k+t+h-2}^{h} + \sum_{h=k-j+1}^{s-t} e_{h,t+h-1}^{h} + e_{s-t+1,s})
\end{align*}
\end{align*}
\]
and again, for $t \leq k \leq s - 1$

$$z_{A}^{t-1,k} = \left( \sum_{h=1}^{t-1} e^{h,k-t+h+1}, 0 \right) + (0, \sum_{h=1}^{k-t+1} e^{hh} + \sum_{h=k-t+2}^{s-t+1} e^{h,t+h-1})$$

Solving the associated systems, taking into account of the values previously computed, one gets that the only components of $\pi$ different from 0 are

$$\pi_{A}^{2t-s-2,t-1} = \pi_{0} - \pi_{11}^{A} - \pi_{1,2t-s-1}^{B} - \pi_{s-t+1,s}^{B}$$

$$\pi_{j,t-1} = \pi_{0} - \pi_{11}^{A} - \pi_{1,j}^{B} - \pi_{s-t+1,s}^{B}$$

$$\pi_{j,k} = \pi_{0} - \pi_{11}^{A} - \pi_{1,2t-s-1}^{B} - \pi_{s-t+1,s}^{B}$$

$$t \leq k \leq s - t + j + 1$$

$$\pi_{i-1,k} = \pi_{0} - \pi_{1,k-t+2}^{A} - \pi_{11}^{B} - \pi_{s-t+1,s}^{B}$$

$$t \leq k \leq s - 1$$

for $j \in A$, $2t-s-1 \leq j \leq t-2$. Substituting them in the expressions of $\pi_{i-1,t-1}$ and $\pi_{i-1,s}$, it follows

$$\pi_{i-1,t-1} = \pi_{i-1,s} = \pi_{0}$$

Then, equalities (4.21)-(4.32) identify all the components of $\pi$ whose value is different from 0. Plugging them into $\pi^{T}z = \pi_{0}$, one concludes that for $k \in T$ such that $s - t + 2 \leq k \leq s$

$$\pi_{i-1,t-1} = \pi_{i-1,s} = \pi_{0}$$

$$\pi_{s-t+1,s} = \pi_{s-t+1,k} = \pi_{0} \quad k \in T, s - t + 2 \leq k \leq s$$

$$\pi_{11}^{B} = -\pi_{0}$$

and all the remaining components equal 0. Then, the thesis follows.

For inequalities (4.20) the proof is analogous.

Again at most one $y$-variable ($x$-variable) having coefficient equal to 1 can take value 1. Further, if $y_{jt} = 0$ ($x_{it} = 0$) but there exists $t' \leq t - 1$ such that $y_{jt'} = 1$ ($x_{it'} = 1$) then the number of remaining time slots is not sufficient to schedule one job of each user and having an assignment variable appearing in the constraints plus all the previous jobs. Finally, if $\sum_{s=1}^{t'} y_{js} = 0$ ($\sum_{i=1}^{s} x_{is} = 0$), then the precedence constraints imply that none of the jobs of user $B$ ($A$) having an assignment variables appearing in the constraint can be scheduled.

The last family of facets we describe has the form

$$x(S_{1}) + y(S_{2}) \leq 2$$
Theorem 4.1.9. Inequalities for $j_i$ and $x_j$, or $y_i$ and $y_j$ (cases (a), (b), (c) of Section 3.2), but also between $x_i$ and $y_j$, precisely when

(d) $j \leq s - i$, $t = s$, or

(e) $j \geq t - i + 1$ and $s \in T$, $j \leq s \leq j + i - 1$.

Consider then the 5-holes on $G$ having the form

\[
H' = \{x_{it}, x_{i,t+1}, x_{i+1,t+2}, y_{t-i,t}, y_{t-i,t+2}\}
\]

\[
H'' = \{y_{js}, y_{j,s+1}, y_{j+1,s+2}, x_{s-j,s}, x_{s-j,s+2}\}
\]

for $i \in A$, $i \leq n_1 - 2$, $t \in T$, $i + 1 \leq t \leq m - 2$, $j \in B$, $j \leq n_2 - 2$ and $s \in T$, $j + 1 \leq s \leq m - 2$.

Hence, inequalities

\[
x_{it} + x_{i,t+1} + x_{i+1,t+2} + y_{t-i,t} + y_{t-i,t+2} \leq 2
\]

\[
y_{js} + y_{j,s+1} + y_{j+1,s+2} + x_{s-j,s} + x_{s-j,s+2} \leq 2
\]

are valid for $P_S$. Let

\[
f'(i, t) = x_{it} + x_{i,t+1} + x_{i+1,t+2} + y_{t-i,t} + y_{t-i,t+2}
\]

\[
f''(j, s) = y_{js} + y_{j,s+1} + y_{j+1,s+2} + x_{s-j,s} + x_{s-j,s+2}
\]

for all $i \in A$, $i \leq n_1 - 2$, $t \in T$, $i + 1 \leq t \leq m - 2$, $j \in B$, $j \leq n_2 - 2$ and $s \in T$, $j + 1 \leq s \leq m - 2$.

**Theorem 4.1.9.** Inequalities

\[
f'(i, t) + \sum_{s=i+1}^{t} x_{i+1,s} + \min\{t+2,n_1\} x_{j,t+2} + \sum_{j=i+3}^{t+1} y_{js} + \sum_{s=j}^{t+1} y_{j,t+2} \leq 2
\]

for $i \in A$, $i \leq n_1 - 2$ and $t \in T$, $i + 1 \leq t \leq m - 2$, and

\[
f''(j, s) + \sum_{h=j+1}^{s} y_{j+1,h} + \min\{s+2,n_2\} y_{h,s+2} + \sum_{h=j+3}^{s+1} x_{ih} + \sum_{h=i}^{s+1} x_{h,s+2} \leq 2
\]

(4.33)

for $j \in B$, $j \leq n_2 - 2$ and $s \in T$, $j + 1 \leq s \leq m - 2$, induce facets of $P_S$. 

Proof. For \( i \in A, i \leq n_1 - 2 \) and \( t \in T, i + 1 \leq t \leq m - 2 \), let

\[
F_{it} = \{ z = (x, y) \in P_S : \ z \text{ verifies inequalities (4.33) with sign '=} \}
\]

and let \( \pi^T z = \pi_0 \) be any hyperplane containing \( F_{it} \). We next show that

\[
\pi^T z = 2\pi_0(f'(i, t) + \sum_{s=i+1}^{t} x_{i+1,s} + \sum_{j=i+3}^{t+i+2} x_{j,t+2} + \sum_{s=j}^{t-i+2} y_{s,t+2} + \sum_{j=t-n_1+2}^{t-i-1} y_{j,t+2})
\]

For more semplicity, choose \( i \in A \) and \( t \in T \) in such way that \( t + 2 = n_1 \) (if it is not, the theorem can be proved in symilar way).

Define

\[
z_{it}^A = (\sum_{h=1}^{i-1} e^{hh} + e^{i+1,t+1} + e^{i+2,t+2}, 0)
\]

and observe that the components whose value is 1 in the above vector correspond to nodes in the conflict graph \( G \) identifying a chord of \( H' \). Solving the system \( \pi^T z_{it}^A = \pi_0 \), it follows

\[
\pi^A_{it} = \pi_0 - \sum_{j=1}^{i-1} \pi_{jj}^A - \pi_{i+1,t+1}^A - \pi_{i+2,t+2}^A
\]

For \( k \in B, 2 \leq k \leq t - i \) and \( s \in T \), define

\[
z_{it}^B = \pi_{it}^A + (0, e^{ls}) \quad i \leq s \leq t - 1
\]

\[
z_{ks}^B = z_{it}^A + (0, \sum_{h=1}^{k-1} e^{h,i+h-1} + e^{ks}) \quad i + k - 1 \leq s \leq t - 1
\]

and solve the associated systems. Exploiting the just obtained value of \( \pi^A_{it} \), one gets

\[
\pi^B_{ks} = 0
\]

for \( k \in B, k \leq t - i \), and \( s \in T, i + k - 1 \leq s \leq t - 1 \).

For \( k \in B, k \geq 2 \), and \( s \in T \) define

\[
z_{it}^B = \pi_{it}^A + (0, e^{ls}) \quad s \geq t + 3
\]

\[
z_{ks}^B = z_{it}^A + (0, \sum_{h=1}^{k-1} e^{h,t+h+2} + e^{ks}) \quad s \geq t + k + 2
\]

and plug the above vectors into \( \pi^T z = \pi_0 \). Solving the associated systems, one obtains

\[
\pi^B_{ks} = 0
\]
for \( k \in B \) and \( s \in T, s \geq t + k + 2 \).

Define again
\[
\mathbf{z}^{t+1}_{i,t} = \left( \sum_{h=1}^{i} \mathbf{e}^{h,t-i+h+1}, 0 \right) + \left( 0, \sum_{h=1}^{t-i-1} \mathbf{e}^{h,t} \right)
\]

Solving \( \pi^T \mathbf{z}^{i,t+1}_A = \pi_0 \), it follows
\[
\pi^A_{i,t+1} = \pi_0 - \sum_{h=1}^{i-1} \pi^A_{h,t-i+h+1} - \sum_{h=1}^{t-i-1} \pi^B_{h,h} - \pi^B_{t-i,t+2}
\]

For \( k \in B, k \geq t - i + 2 \), define
\[
\mathbf{z}^{i_{t-i+1},s}_B = \mathbf{z}^{i,t+1}_A + \left( \mathbf{e}^{i_{t-i+1},s}, 0 \right) \quad t + 3 \leq s \leq 2t - i + 2
\]
\[
\mathbf{z}^{k_{s,t+1},s}_B = \mathbf{z}^{i,t+1}_A + \left( 0, \sum_{h=1}^{k+i-1} \mathbf{e}^{i_{t-i+h},t+t+2} + \mathbf{e}^{k,s} \right) \quad k + i + 2 \leq s \leq t + k + 1
\]

and solve the associated systems, exploiting also the value of \( \pi^A_{i,t+1} \). It follows
\[
\pi^B_{k,s} = 0
\]

for \( k \in B, k \geq t - i + 1 \), and \( s \in T, k + i + 2 \leq s \leq t + k + 1 \).

For \( k \in A, k \geq i + 2 \), and \( s \in T \) define
\[
\mathbf{z}^{i+1,s}_A = \mathbf{z}^{i,t+1}_A + \left( \mathbf{e}^{i+1,s}, 0 \right) \quad s \geq t + 3
\]
\[
\mathbf{z}^{k_{s,t+1},s}_A = \mathbf{z}^{i,t+1}_A + \left( \sum_{h=1}^{k-i-1} \mathbf{e}^{i+h,t+t+2} + \mathbf{e}^{k,s}, 0 \right) \quad s \geq t + k - i + 2
\]

Solving the associated systems, one gets
\[
\pi^A_{k,s} = 0
\]

for \( k \in A, k \geq i + 1 \) and \( s \in T, s \geq t + k - i + 2 \).

Defining again
\[
\mathbf{z}^{i+2,t+2}_A = \left( \sum_{h=1}^{i} \mathbf{e}^{h,h} + \sum_{h=1}^{t-i} \mathbf{e}^{i+1,t+1} + \mathbf{e}^{i+2,t+2}, 0 \right) + \left( 0, \sum_{h=1}^{t-i} \mathbf{e}^{h,i+h} \right)
\]
\[
\mathbf{z}^{t-i,t+2}_B = \left( \sum_{h=1}^{t-i} \mathbf{e}^{h,t-i+h-1} + \mathbf{e}^{t,i}, 0 \right) + \left( 0, \sum_{h=1}^{t-i-1} \mathbf{e}^{h,h} + \mathbf{e}^{t-i,t+2} \right)
\]
and solving the associated systems, one obtains

\[
\begin{align*}
\pi^A_{i+2,t+2} &= \pi_0 - \sum_{h=1}^{i} \pi^A_{hh} - \pi^A_{i+1,t+1} - \sum_{h=1}^{t-i} \pi^B_{h,i+h} \\
\pi^B_{t-i,t+2} &= \pi_0 - \sum_{h=1}^{t-i-1} \pi^B_{hh} - \sum_{h=1}^{i-1} \pi^A_{h,t-i+h-1} - \pi^A_{it}
\end{align*}
\]

For \( k \in A, k \geq i + 2 \), define

\[
\begin{align*}
z^A_{i+1,t+1} &= z^B_{i+1,t+2} + (e^{i+1,t+1}, 0) \\
z^A_{k,t+k-i+1} &= z^B_{i+2} + (e^{i+1,t+1} + \sum_{h=2}^{k-i} e^{h,t+h+1}, 0)
\end{align*}
\]

and solve the associated systems in order to get

\[
\pi^A_{k,t+k-i+1} = 0
\]

for all \( k \in A, k \geq i + 2 \).

Defining

\[
\begin{align*}
z^A_{i+1,t+2} &= z^B_{t-i,t} + (e(i+1,t+2), 0) \\
z^B_{t-i,t} &= \left(\sum_{h=1}^{i-1} e^{h,t-i+h-1} + e^{i+1,t+1}, 0\right) + \left(0, \sum_{h=1}^{t-i-1} e^{h,t} + e^{t-i,t}\right)
\end{align*}
\]

from the associated systems, one gets

\[
\begin{align*}
\pi^A_{i+1,t+2} &= 0 \\
\pi^B_{t-i,t} &= \pi_0 - \sum_{h=1}^{t-i-1} \pi^B_{hh} - \sum_{h=1}^{i-1} \pi^A_{h,t-i+h-1} - \pi^A_{it+1}
\end{align*}
\]

For \( k \in B, k \geq t - i + 2 \) define

\[
\begin{align*}
z^B_{t-i+1,t+2} &= z^B_{t-i,t} + (0, e^{t-i+1,t+2}) \\
z^B_{k,k+i+1} &= z^B_{t-i,t} + (0, \sum_{h=1}^{k+i-t} e^{t-i+h,t+h+1})
\end{align*}
\]

and solve the associated systems. It follows

\[
\pi^B_{k,k+i+1} = 0
\]
Again, for \( s \in T, i + 1 \leq s \leq t \) defining

\[
z_{A}^{i+1,s} = \left( \sum_{h=1}^{i} e_{h,s-i+h-1} + e_{i+1,s} + e_{i+2,t+2}, 0 \right)
\]

and solving \( \pi^{T}z_{A}^{i+1,s} = \pi_{0} \), one obtains

\[
\pi_{i+1,s}^{A} = \pi_{0} - \sum_{h=1}^{i} \pi_{h,s-i+h-1}^{A} - \pi_{i+2,t+2}^{A}
\]

For \( k \in B, 2 \leq k \leq t - i - 1 \), and \( s \in T, s \leq t - i - 1 \) define

\[
z_{B}^{1s} = z_{A}^{i+1,i+s+1} + (0, e_{1s})
\]

\[
z_{B}^{ks} = z_{A}^{i+1,t} + (0, \sum_{h=1}^{k-1} e_{hh} + e_{ks})
\]

Solving the systems \( \pi^{T}z_{B}^{1s} = \pi_{0} \) and \( \pi^{T}z_{B}^{ks} = \pi_{0} \), it follows

\[
\pi_{B}(k, s) = 0
\]

for \( k \in B, k \leq t - i - 1 \), and \( s \in T, s \leq t - i - 1 \).

For \( k \in B, 2 \leq k \leq t - i \) and \( s \in T, s = t, t+1 \) define

\[
z_{B}^{1s} = z_{A}^{i+1,i+1} + (0, e_{1s})
\]

\[
z_{B}^{ks} = z_{A}^{i+1,i+k} + (0, \sum_{h=1}^{k-1} e_{hh} + e_{ks})
\]

and solve the associated systems in order to obtain

\[
\pi_{kd}^{B} = 0
\]

for \( k \in B, k \leq t - i \), and \( s \in T, s = t, t+1 \).

For \( s \in T, t - i + 3 \leq s \leq t \), define

\[
z_{B}^{t-i+1,t-i+1} = \left( \sum_{h=1}^{i} e_{h,t-i+h+1} + e_{t-i+1,t-i+1}, 0 + \sum_{h=1}^{t-i} e_{hh} + e_{t-i+1,t-i+1} \right)
\]

\[
z_{B}^{t-i+1,t-i+2} = \left( \sum_{h=1}^{i} e_{h,t-i+h+1} + e_{t-i+1,t-i+2}, 0 + \sum_{h=1}^{t-i} e_{hh} + e_{t-i+1,t-i+2} \right)
\]

\[
z_{B}^{t-i+1,s} = \left( \sum_{h=i}^{t-i+s-1} e_{h,t-i+h} + \sum_{h=s+i-t}^{i} e_{h,t-i+h+1} + e_{t-i+1,s}, 0 + \sum_{h=1}^{t-i} e_{hh} + e_{t-i+1,s} \right)
\]
and solve the associated systems, obtaining

\[
\pi_{t-i+1,t-i+1}^B = \pi_0 - \sum_{h=1}^{t-i} \pi_{h,t-i+1}^B - \sum_{h=1}^{i} \pi_{h,t-i+h+1}^A
\]

\[
\pi_{t-i+1,t-i+2}^B = \pi_0 - \sum_{h=1}^{t-i} \pi_{h,t-i+2}^B - \sum_{h=1}^{t-i} \pi_{h,t-i+h+1}^A
\]

\[
\pi_{t-i+1,s}^B = \pi_0 - \sum_{h=1}^{t-i} \pi_{h,s+i-t-1}^B - \sum_{h=1}^{i} \pi_{h,s+i-t}^A - \sum_{h=1}^{s+i-t} \pi_{h,t-i+h+1}^A
\]

Now, defining

\[
z_{t-i+1,t+1}^B = (0, \sum_{h=1}^{t-i+1} e_h^{i+h})
\]

and solving \(\pi^T z_{t-i+1,t+1}^B = \pi_0\), one gets

\[
\pi_{t-i+1,t+1}^B = \pi_0 - \sum_{h=1}^{t-i} \pi_{h,i+h}^B
\]

and again, for \(k \in A, 2 \leq k \leq i - 1\), and \(s \in T, s \leq i\), define

\[
z_{A}^{1s} = z_{B}^{t-i+1,t+1} + (e_1^s, 0)
\]

\[
z_{A}^{ks} = z_{B}^{t-i+1,t+1} + \left(\sum_{h=1}^{k-1} e_{hh}, e_{k}^s, 0\right)
\]

\[
z_{A}^{ii} = \left(\sum_{h=1}^{i} e_{hh}, 0\right)
\]

and solve the associated systems. It follows

\[
\pi_{A}^{ks} = 0
\]

for \(k \in A, k \leq i - 1\), and \(s \in T, s \leq i\). Using the same definition as for the above vectors also for \(k \in A, 2 \leq k \leq i\), and \(s \in T, s \geq t+2\), by the associated systems it follows

\[
\pi_{A}^{ks} = 0
\]

For \(s \in T, t-i+2 \leq s \leq t+1\), define

\[
z_{B}^{t-i+2,s} = (0, \sum_{h=1}^{t-i+2} e_{h,s-t+i-2})
\]
and solve \( \pi^T z^{t-i+2,s}_B = \pi_0 \) in order to have
\[
\pi^{B}_{t-i+2,s} = \pi_0 - \sum_{h=1}^{t-i+1} \pi^{B}_{h, s-t+i+h-2}
\]

and again, for \( k \in A, 2 \leq k \leq i - 1 \), and \( s \in T \)
\[
z^{1s}_A = z^{t-i+2,t-i+2}_B + (e^{1s}, 0) \quad t - i + 3 \leq s \leq t + 1
\]
\[
z^{ks}_A = z^{t-i+2,t-i+2}_B + (e^{h,t-i+h+2} + e^{ks}, 0) \quad t - i + k + 2 \leq s \leq t + 1
\]

solving the associated systems, one gets
\[
\pi^{A}_{ks} = 0
\]

For \( k \in B, 4 \leq k \leq n_2 + i - t \), and \( s \in T \), defining
\[
z^{t-i+3,s}_B = z^{t-i+2,t-i+2}_B + (0, e^{t-i+3,s}) \quad t - i + 3 \leq s \leq t + 3
\]
\[
z^{t-i+k,s}_B = z^{t-i+2,t-i+2}_B + (\sum_{h=1}^{t-i+k-1} e^{h}, 0, e^{t-i+k,s}) \quad t - i + k \leq s \leq t + k
\]

and solving the associated systems, it follows that all the components corresponding to the above vectors equal 0.

Again, for \( j \in A, i + 3 \leq j \leq n_1 \), and \( k \in B, k \leq t - i - 1 \), define
\[
z^{j,t+2}_A = (\sum_{h=1}^{i} e^{h} + \sum_{h=1}^{j-i} e^{i+h,t+i-j+h+2}, 0) \quad (4.35)
\]
\[
z^{k,t+2}_B = (\sum_{h=1}^{i} e^{h,t-i+h}, 0) + (0, \sum_{h=1}^{k-1} e^{h} + e^{k,t+2})
\]

and, solving the associated systems, one gets
\[
\pi^{A}_{j,t+2} = \pi_0 - \sum_{h=1}^{i} \pi^{A}_{h,h} - \sum_{h=1}^{j-i-1} \pi^{A}_{h, t+i-j+h+2}
\]
\[
\pi^{B}_{k,t+2} = \pi_0 - \sum_{h=1}^{k-1} \pi^{B}_{h,h} - \sum_{h=1}^{i} \pi^{A}_{h, t+i+h}
\]

In the following, group by different families \( \mathcal{F}_i \) the vectors allowing to compute the values of the remaining components of \( \pi \).
1. For \( k \in A, k \leq i - 1 \), and \( s \in T, i + 1 \leq s \leq t - i + k \), family \( \mathcal{F}_1 \) contains

\[
\begin{align*}
\mathbf{z}_{A}^{i,s} &= \left( \sum_{h=1}^{i} \mathbf{e}_{h,s-i+h}, 0 \right) + \left( 0, \sum_{h=1}^{s-i} \mathbf{e}_{h} + \sum_{h=s-i+1}^{t-i+1} \mathbf{e}_{h,i+h} \right) \\
&\quad \quad i + 1 \leq s \leq t - 1 \\
\mathbf{z}_{A}^{k,s} &= \left( \sum_{h=1}^{k} \mathbf{e}_{h,s-k+h}, 0 \right) + \left( 0, \sum_{h=1}^{s-k} \mathbf{e}_{h} + \sum_{h=s-k+1}^{t-i+1} \mathbf{e}_{h,i+h} \right) \\
\mathbf{z}_{A}^{k,t-i+k+1} &= \left( \sum_{h=1}^{k} \mathbf{e}_{h,t-i+k+1}, 0 \right) + \left( 0, \sum_{h=1}^{t-i+1} \mathbf{e}_{h} + \mathbf{e}_{t-i+2,t+1} \right)
\end{align*}
\]

Solving the associated systems, from all the components of \( \pi \) previously calculated, it follows

\[
\pi_{A}^{s} = 0
\]

2. For \( k \in A, k \geq i + 2 \), and \( s \in T, k + 1 \leq s \leq t + 1 \), family \( \mathcal{F}_2 \) contains

\[
\begin{align*}
\mathbf{z}_{A}^{k,k} &= \left( \sum_{h=1}^{k} \mathbf{e}_{h}, 0 \right) + \left( 0, \sum_{h=1}^{t-i} \mathbf{e}_{h,k+h} \right) \\
\mathbf{z}_{A}^{k,s} &= \left( \sum_{h=1}^{k-1} \mathbf{e}_{h} + \mathbf{e}_{k,s}, 0 \right) + \left( 0, \sum_{h=1}^{s-k} \mathbf{e}_{h,k-1+h} + \mathbf{e}_{s-k+1,s+1} \right)
\end{align*}
\]

Solving the associated systems, one obtains

\[
\pi_{k,s}^{A} = \pi_{0} - \pi_{t+i+1,i+1}^{A} - \pi_{t-i,t+2}^{B}
\]

3. For \( s \in T \), family \( \mathcal{F}_3 \) contains vectors

\[
\begin{align*}
\mathbf{z}_{B}^{t-i-1,s} &= \left( 0, \sum_{h=1}^{t-i-2} \mathbf{e}_{h,b+1} + \mathbf{e}_{t-i-1,s}^{t-i,t} + \mathbf{e}_{t-i+1,t+1} \right) \\
&\quad \quad t - i \leq s \leq t - 3 \\
\mathbf{z}_{B}^{t-i,s} &= \left( 0, \sum_{h=1}^{t-i-1} \mathbf{e}_{h} + \mathbf{e}_{t-i,s}^{t-i,1.t} + \mathbf{e}_{t-i+1,t+1} + \mathbf{e}_{t-i+2,t+1} \right) \\
&\quad \quad t - i \leq s \leq t - 2
\end{align*}
\]

Plugging them into \( \pi^{T} \mathbf{z} = \pi_{0} \) and solving the systems, one has that all the resulting components of \( \pi \) equal 0.

4. For \( k \in B, 3 \leq k \leq t - i \), and \( s \in T, t + 3 \leq s \leq 2t - i - k + 1 \), family \( \mathcal{F}_4 \) contains

\[
\mathbf{z}_{B}^{k,s} = \left( \sum_{h=1}^{i+3} \mathbf{e}_{h,t-i+h+1}, 0 \right) + \left( 0, \sum_{h=1}^{k-1} \mathbf{e}_{h} + \mathbf{e}_{ks} \right)
\]
Solving the associated systems, one obtains that components $\pi^B_{ks}$ equal 0.

5. For $k \in B$, $k \leq t - i - 2$, and $s \in T$, $t - i \leq s \leq i - 1$, family $\mathcal{F}_5$ contains

$$z^B_{ks} = \left( \sum_{h=1}^{i-1} e^{hh} + e^{it} + e^{j+1,t+1} + e^{i+2,t+2}, 0 \right) + \left( 0, \sum_{h=1}^{k} e^{h,s-k+h} \right)$$

Solving the associated systems, it follows that all the components of $\pi$ corresponding to the above vectors equal 0.

6. Finally, for $k \in A$, $k \geq i + 4$, and $s \in T$, $t + 3 \leq s \leq t + k - i$, family $\mathcal{F}_6$ contains

$$z^A_{ks} = \left( \sum_{h=1}^{k-1} e^{hh} + e^{ks}, 0 \right)$$

By the associated systems, it follows that all the components of $\pi$ corresponding to the above vectors equal 0.

At this point, solving the system containing all the equalities associated with components of $\pi$ different from 0 but (4.35), one obtains that these components equal $\frac{2}{3}$. Replacing them in (4.35) and in the equalities in family $\mathcal{F}_2$, it follows that all the remaining components equal 0 and the theorem is proved.

For inequalities (4.34) the proof is analogous. □

Finding the general form of all facet-defining inequalities corresponding to maximal cliques of the conflict graph $G$ and those obtained by lifting odd-holes inequalities is an open problem. We observe that by Theorem 3.2.6 no subgraph of $G$ induced by vertices corresponding to $x$ (the $y$) variables only contains an odd hole.

4.2 SP$_2$

For the shuffling problem, a time-indexed formulation is obtained from (4.1)-(4.10) by replacing first and second set of inequality constraints by equalities. Note that in this case $x_{it}$ ($y_{jt}$) is defined only for $i \in A$ ($j \in B$) and $t \in T$, $i \leq t \leq i + n_2$ ($j \leq t \leq j + n_1$).

As usual, denote as $P_S$ the convex hull of feasible solutions to SP$_2$. Some facet-inducing inequalities of $P_S$ can be found through the same technique as in section 3.3. Associate a variable $z_{jt}$ with each $f(j, t)$ in formula (3.9), and consider the resulting linear program
4.3 CSP

\[ \min z_{n2m} \quad (4.36) \]
\[
z_{0j} - z_{0,j-1} \geq a_{jj} \quad j \in B \\
z_{jt} - z_{j,t-1} \geq a_{t-j,t} \quad j \in B, \ t > j \\
z_{jt} - z_{j-1,t-1} \geq b_{jt} \quad j \in B, \ t \geq j \\
\]

Let \( w_{t-j,t} \) and \( v_{jt} \) be the dual variables associated with program (4.36). The dual is
\[
\max \sum_{j \in J} \sum_{t \geq j} (a_{t-j,t} w_{t-j,t} + b_{jt} v_{jt}) \quad (4.37)
\]
\[
w_{jj} - v_{1,j+1} - w_{j+1,j+1} = 0 \quad j \in B \\
v_{jt} + w_{t-j,t} - v_{j+1,t+1} - w_{t-j+1,t+1} = 0 \quad j \in B - \{n_2\}, \ t \geq j \\
v_{nzt} + w_{t-nz,t} - w_{t-nz+1,t+1} = 0 \quad t \leq n - 1 \\
v_{nzn} + w_{n-nz,n} = 1 \\
v_{jt}, w_{t-j,t} \geq 0 \quad j \in B, \ t \geq j \\
\]
Comparing the objective function of (4.1) to that of (4.37), one obtains \( w_{it} = x_{it} \) for all \( i \in A \), and \( v_{jt} = y_{jt} \) for all \( j \in B \). Observe that the solutions to SP\(_2\) are therefore in a one-to-one correspondence with \((s, t)\)-paths of a directed acyclic grid \( G \) constructed as follows (see Figure 4.3)

- there exists a node \((j, t)\) for each \( j \in B \cup \{0\} \) and \( j \leq t \leq j + n_1 \);
- \( s = (0, 0), t = (n_2, m) \);
- the horizontal edges of the graph are of the form \((j - 1, t - 1) \to (j, t)\) and are weighted by \( b_{jt} \);
- the vertical edges are of the form \((j, t - 1) \to (j, t)\) and are weighted by \( a_{t-j, t} \).

Being \( G \) acyclic, the convex hull of its \((s, t)\)-paths is \( \{z \in \mathbb{R}^d | Ez = e_t, z \geq 0\} \), where \( E \) is obtained by removing row \( s \) from the node-arc incidence matrix of \( G \) and \( d \) is equal to \((m + 1) \sum_{k=1}^2 n_k - \sum_{k=1}^2 \frac{n_k(n_k+1)}{2} \).

4.3 CSP\(_2\)

Problem CSP\(_2\) generalizes SP\(_2\) to the case in which \( m > n_1 + n_2 \). Observe that, in this case variable \( x_{it} \) (\( y_{js} \)) is defined if and only if \( i \leq t \leq i + m - n_1 \) (\( j \leq s \leq j + m - n_2 \)). Consider an optimal assignment of exactly \( i \) jobs in \( A \)
and of exactly \( j \) jobs in \( B \) to \( t \geq i + j \) slots and denote as \( z^j_{is} \) the corresponding utility. Writing the constraints derived by formula (3.8) one gets the linear program

\[
\begin{align*}
\min & \quad z_{n1m}^{n2} \\
z^j_{is} - z^j_{i-1,s-1} & \geq a_{is} \quad i \in A, j \in B \cup \{0\}, i + j \leq s \leq i + m - n_1 \\
z^j_{is} - z^{j-1}_{i,s-1} & \geq b_{js} \quad j \in B, i \in A \cup \{0\}, i + j \leq s \leq j + m - n_2 \\
z^j_{is} - z^j_{i,s-1} & \geq 0 \quad i \in A, j \in B \cup \{0\}, i + j \leq s \leq \max\{i + m - n_1, j + m - n_2\}
\end{align*}
\]

Let \( w^j_{is}, v^j_{is} \) and \( u^j_{is} \) be the dual variables associated with the constraints in program. For \( i \in A, j \in B \) and \( s \geq i + j \), the dual is
4.3. CSP$_2$

\[
\max \left( \sum_{i \in A} \sum_{s \geq i} a_{is} \sum_{j=0}^{s-i} w_{is}^j + \sum_{j \in B} \sum_{s \geq j} b_{js} \sum_{i=0}^{s-j} v_{is}^j \right) \quad (4.39)
\]

\[
v_{is}^j + w_{is}^j + u_{i,s-1}^j - v_{i+1,s+1}^j - u_{is}^j = 0 \quad i \neq n_1 \quad (4.40)
\]

\[
v_{is}^j + w_{is}^j + u_{i,s-1}^j - v_{i+1,s+1}^j - u_{is}^j = 0 \quad j \neq n_2 \quad (4.41)
\]

\[
v_{n_1,s}^{n_2} + w_{n_1,s}^{n_2} + u_{n_1,s-1}^{n_2} - u_{n_1}^{n_2} = 0 \quad s \leq m - 1 \quad (4.42)
\]

\[
v_{n_1 m}^{n_2} + w_{n_1 m}^{n_2} + u_{n_1,m-1}^{n_2} = 1 \quad (4.43)
\]

\[
w_{is}^j, v_{is}^j, u_{is}^j \geq 0 \quad (4.44)
\]

Comparing to each other the objective functions of (4.1) to that (4.39), one obtains

\[
\sum_{j=0}^{t-i} w_{it}^j = x_{it} \quad t \leq i + m - n_1 \quad \text{and} \quad \sum_{i=0}^{t-j} v_{it}^j = y_{jt} \quad j \leq j + m - n_2
\]

for each $i \in A$ and $j \in B$.

Similarly to Section 4.2, a partial description of the convex hull $P_S$ of CSP$_2$ is then derived by projecting the polyhedron of program (4.39) into the subspace $(w, v, z) = (w, v, 0)$.

**Projection 1**

1. Rewriting the first dual constraints for job $n_2 \in B$

\[
w_{is}^{n_2} + v_{is}^{n_2} + u_{i,s-1}^{n_2} - w_{i+1,s+1}^{n_2} - u_{is}^{n_2} = 0
\]

and summing them up for $s \in [n_2, m]$, and $i \in [0, s - n_2]$ one obtains

\[
\sum_{s=n_2}^{m} \sum_{i=0}^{s-n_2} v_{is}^{n_2} = 1
\]

2. Rewriting the first dual constraints for each $j \in [1, n_2 - 1]$

\[
w_{is}^{j} + v_{is}^{j} + u_{i,s-1}^{j} - w_{i+1,s+1}^{j} - u_{is}^{j} = 0
\]

and summing them up for $s \in [j, j + m - n_2]$ and $i \in [0, s - j]$ one gets

\[
\sum_{s=j}^{j+m-n_2} \sum_{i=0}^{s-j} v_{is}^{j} = \sum_{s=j+1}^{j+1+m-n_2} \sum_{i=0}^{s-j-1} v_{is}^{j+1}
\]
Putting together the last equation and the one obtained at Step 1, one ends up with
\[ \sum_{s \geq j}^{s-i} \sum_{i=0}^{j} v_{is} = 1. \]

Symmetrically, for each \( i \in A \) one obtains
\[ \sum_{s \geq i}^{s-i} \sum_{j=0}^{i} w_{is} = 1 \]

From the correspondence between original and dual variables, it follows
\[ \sum_{s \geq j}^{y_{js}} = 1 \quad \forall j \in B \]
\[ \sum_{s \geq i}^{x_{is}} = 1 \quad \forall i \in A \]

i.e. the assignment constraints on the jobs.

**Projection 2**

1. Suppose that \( n_2 > n_1 \), rewrite the first dual constraints for slot \( t \in T, t \geq n_2 + 1 \)
\[ u_{i,t}^h + v_{i,t}^h + u_{i,t-1}^h - w_{i,t+1,t+1}^h - v_{i,t+1}^h - u_{it}^h = 0 \]
and sum them up for \( i \in A, i \in [0, \min\{t, n_1\}], \) and \( h \in [0, t - i] \). The resulting equation is
\[ \sum_{i=\max\{1,t-m+n_1\}}^{\min\{t,n_1\}} x_{it} + \sum_{j=\max\{1,t-m+n_2\}}^{\min\{t,n_2\}} y_{jt} + \sum_{j=0}^{\min\{t-1-j,n_1\}} \sum_{i=0}^{n_2} u_{jt}^{j+i-1} + \sum_{j=\max\{1,t+1-m+n_2\}}^{\min\{t+1,n_2\}} y_{j,t+1} + \sum_{j=0}^{\min\{t-j,n_1\}} \sum_{i=0}^{n_2} u_{j,t}^{j+i-1} = \]
2. In particular, rewriting the equation of Step 1. for \( t = m \) it follows
\[ x_{n_1m} + y_{n_2m} + \sum_{j=0}^{n_2} \sum_{i=0}^{\min\{m-j,n_1\}} u_{i,n-1}^{j} = 1 \]
and one concludes
\[ \min\{t,n_1\} \sum_{i=\max\{1,t-m+n_1\}} x_{it} + \min\{t,n_2\} \sum_{j=\max\{1,t-m+n_2\}} y_{jt} \leq 1 \] (4.45)

for each \( t \in T \), \( t \geq n_2 + 1 \), i.e. a particular case of the assignment constraints on the slots. Observe that, if \( n_1 \geq n_2 \) one uses a similar projection procedure for \( t \in T \), \( t \geq n_1 + 1 \).

**Projection 3**

1. Rewriting the first dual constraints for job \( n_2 \in B \)

\[ u^{n_2}_{it} + v^{n_2}_{it} + u^{n_2}_{i,t-1} - w^{n_2}_{i+1,t+1} - u^{n_2}_{it} = \alpha \]

and summing them up for \( t \in [n_2 + 1, m] \), and \( i \in [0, t - n_2] \) it follows

\[ \sum_{t \geq n_2 + 1} \sum_{i=0}^{t-n_2} v_{it}^{n_2} = 1 - (w_{1,n_2+1}^{n_1} + u_{0n_2}^{n_2}) \]

i.e.

\[ \sum_{t \geq n_2 + 1} y_{nt} \leq 1. \] (4.46)

Simmetrically,

\[ \sum_{t \geq n_1 + 1} x_{nt} \leq 1. \] (4.47)

2. For each \( j \in B \), \( j \neq n_2 \), rewrite the first dual constraints

\[ w^j_{it} + v^j_{it} + u^j_{i,t-1} - w^j_{i+1,t+1} - v^j_{i,t+1} - u^j_{it} = \alpha \]

and summing them up for \( t \in [j + h, m] \), for each \( h = 1, \ldots, m - j \), and \( i \in [0, t - j] \) it follows

\[ \sum_{t \geq j+h} \sum_{i=0}^{t-j} v_{it}^j = \]

\[ \sum_{t \geq j+h+1} \sum_{i=0}^{t-j-1} v_{it}^j + \sum_{i=\min\{j,n_1\}}^{n_1} (w_{i,j+m-n_2+1}^j + u_{i,j+m-n_2}^j) - (\sum_{i=1}^{\min\{h,n_1\}} w_{i,j+h}^j + \sum_{i=0}^{\min\{h-1,n_1\}} u_{i,j+h}^j) \]

Observe that
• If \( j \geq n_1 \), then \( \sum_{i=\min\{j,n_1\}}^{n_1} (w_{i,j+m-n_2+1}^j + u_{i,j+m-n_2}^j) = 0 \) and one concludes

\[
\sum_{t \geq j+h} \sum_{i=0}^{t-j} v_{it}^j \leq \sum_{t \geq j+h+1} \sum_{i=0}^{t-j-1} v_{it}^{j+1}.
\]

• If \( j < n_1 \), then variables \( w_{i,j+m-n_2+1}^j \) and \( u_{i,j+m-n_2}^j \) are defined if and only if \( j + m - n_2 + 1 \geq i + m - n_1 \). Since \( i + m - n_1 \geq j + m - n_1 \), it follows that the above variables are defined if and only if \( n_1 \leq n_2 - 1 \). Otherwise, one again gets

\[
\sum_{t \geq j+h} \sum_{i=0}^{t-j} v_{it}^j \leq \sum_{t \geq j+h+1} \sum_{i=0}^{t-j-1} v_{it}^{j+1}.
\]

3. Suppose \( n_1 \leq n_2 - 1 \). The first dual constraints for job \( n_1 \in A \) and slot \( m - n_2 + n_1 + 1 \in T \) is

\[
w_{n_1,m-n_2+n_1+1}^j + u_{n_1,m-n_2+n_1}^j = u_{n_1,m-n_2+n_1+1}^j
\]

for each \( j < n_2 \). Considering the same constraints for \( n_1 \in A \) and \( s \in T \) and summing up for \( s \in [m - n_2 + n_1 + 2, m] \), it follows

\[
w_{n_1,m-n_2+n_1+1}^j = - \sum_{s=m-n_2+n_1+2}^{m} w_{n_1s}^j \leq 0.
\]

4. Rewriting the first constraints for each slot \( s \in [j + m - n_2 + 1, m - n_2 + n_1 + 1] \)

\[
w_{is}^j + u_{is-1}^j - w_{i+1,s+1}^j + u_{i+1,s}^j = 0
\]

and summing for \( i \in \min\{j, n_1\}, n_1 \}, one gets

\[
\sum_{i=\min\{j,n_1\}}^{n_1} (w_{is}^j + u_{is-1}^j) = \sum_{i=\min\{j+1,n_1\}}^{n_1} (w_{is+1}^j + u_{is}^j) \quad \forall j \in B, j \neq n_2
\]

and concludes that

\[
\sum_{i=\min\{j,n_1\}}^{n_1} (w_{i,j+m-n_2+1}^j + u_{i,j+m-n_2}^j) = u_{n_1,m-n_2+n_1+1}^j \leq 0.
\]

Then, for each \( j \in B, j \neq n_2 \), it follows

\[
\sum_{t \geq j+h} \sum_{i=0}^{t-j} v_{it}^j \leq \sum_{t \geq j+h+1} \sum_{i=0}^{t-j-1} v_{it}^{j+1}.
\]
4.3. CSP

i.e.
\[
\sum_{t \geq j+h} y_{jt} \leq \sum_{t \geq j+h+1} y_{jt} \quad \forall j \in B, \ j \neq n_2, \ \forall h = 1, \ldots, m - j. \quad (4.48)
\]

which correspond to a particular case of the precedence constraints.

Symmetrically
\[
\sum_{t \geq i+h} x_{it} \leq \sum_{t \geq i+h+1} x_{it} \quad \forall i \in A, \ i \neq n_1, \ \forall h = 1, \ldots, m - i. \quad (4.49)
\]

We conclude that

**Theorem 4.3.1.** Inequalities (4.45), (4.48), (4.49) and the job assignment constraints induce facets of $P_S$. 
4 Polyhedral Study: the case of two users
Chapter 5

Conclusions

Studying single-machine scheduling problem, with or without precedence constraints, is a fundamental topic investigated in a wide range of fields, especially in Operations Research. Depending on the particular scenario, the problem can be conveniently formulated and several solution algorithms can be developed.

In this work, we have examined two different versions of a competitive single-machine scheduling problem (multi-criteria, single decision-maker) with like-chain precedence constraints and provided both a computational complexity investigation and a polyhedral study.

In the first version, we have emphasized the competitive aspects of the model. We have considered a two-user competitive scheduling problem concerning the allocation of two disjoint sets of unit jobs to one machine so that either the worst utility is optimized (M-FCFS Problem) or the utility of one user is maximized while a minimum satisfaction level is guaranteed to the other (β-FCFS Problem). We have proved that, for separable and regular objective functions of the jobs completion times the problems are NP-hard. However, we have proposed a lagrangian heuristic able to cope with a several real-time requirements. The lagrangian dual associated with the relaxation has been solved by a simple binary search. Further, in the case in which the solution obtained by the heuristic is not-feasible for the original problems, we have developed a very fast crossover-algorithm to adjust the current solution. The peculiar structure of data and feasible schedules allowed to obtain a very good performance of the algorithm, compared to a greedy-like heuristic on a set of practical instances.

The main contribution of this combinatorial study consists in proposing a fast lagrangian algorithm able to compute good solutions for competitive scheduling problems with real-time requirements since, in general, a lagrangian relaxation is not a suitable tool to solve problems with time restrictions.
A polyhedral study has been provided for the second version of the problem, where we have optimized the sum of the total utility of each user in the system. Unlike the previous problem, this case is solvable in polynomial time and we have described a dynamic programming solution algorithm. In particular, we have distinguished between two further subproblems depending on the fact that one or two users are in the system, and investigated some mathematical properties of them.

For the case of a single user, we have completely described the polyhedron defined as the convex hull of all feasible solutions by showing that each vertex is integer. The facets describing the polyhedron have been obtained by two different techniques, depending on whether optimal schedules must or may be complete: in the first case (CSP$_1$ Problem), since the polyhedron of the convex hull is not full-dimensional we have used a particular technique based on a reformulation of the dynamic recursion formula; instead, in the second case (PSP$_1$ Problem), we have proved that the polyhedron is full-dimensional and used a well known standard technique.

For the case of two users, we have computed a considerable number of facet-defining inequalities and partially described the polyhedron of the convex hull. As for one user, the polyhedron is full-dimensional when optimal schedules are partial (PSP$_2$ Problem), and not full-dimensional when optimal schedules are complete (CSP$_2$ and SP$_2$ Problems). Then, we have used both the techniques just named in order to obtain the facets.

The main contribution of this polyhedral study consists in including the precedence constraints in the time-indexed formulation of a single-machine scheduling problem and having a description of the polytope defined as the convex hull of feasible solutions. In fact, before this work a similar polyhedron was investigated but either precedence relations were not considered or the original problem was formulated in terms of completion times instead of time-indexed variables. The fact that this description is complete when only one user is in the system is a very important result which made easier the study for two users.

At present, further research should aim at using the facet-defining inequalities obtained from the polyhedral study to improve the bound provided by linear relaxation of the minimum satisfaction constraint (β-FCFS Problem). Our idea consists in adding valid inequalities, and studying the computational features of both linear and lagrangian relaxations with the aim of devising an efficient implicit enumeration algorithm.
Bibliography


