Tilburg University

Step out - Step in Sequencing Games

Musegaas, M.; Borm, P.E.M.; Quant, M.

Publication date:
2014

Document Version
Early version, also known as pre-print

[Link to publication in Tilburg University Research Portal](#)

Citation for published version (APA):
Musegaas, M., Borm, P. E. M., & Quant, M. (2014). *Step out - Step in Sequencing Games*. (CentER Discussion Paper; Vol. 2014-070). CentER, Center for Economic Research.

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

Take down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

No. 2014-070

STEP OUT - STEP IN SEQUENCING GAMES

By

M. Musegaas, P.E.M. Borm, M. Quant

14 November, 2014

ISSN 0924-7815
ISSN 2213-9532

Step out - Step in Sequencing Games

M. Musegaas ^{*†} P.E.M. Borm ^{*} M. Quant^{*}

November 14, 2014

Abstract

In this paper a new class of relaxed sequencing games is introduced: the class of Step out - Step in sequencing games. In this relaxation any player within a coalition is allowed to step out from his position in the processing order and to step in at any position later in the processing order. Providing an upper bound on the values of the coalitions we show that every Step out - Step in sequencing game has a non-empty core. This upper bound is a sufficient condition for a sequencing game to have a non-empty core. Moreover, this paper provides a polynomial time algorithm to determine the coalitional values of Step out - Step in sequencing games.

Keywords: cooperative game theory, sequencing games, core

JEL classification number: C71, C44

1 Introduction

In this paper one-machine sequencing situations are considered with a queue of players in front of a single machine, each with one job to be processed. Such a situation specifies for each player the processing time, time the machine takes to process the corresponding job of this player. In addition, it is assumed that each player has a linear cost function specified by an individual cost parameter. Moreover, there are no restrictive assumptions as due dates, ready times or precedence constraints imposed on the jobs. To minimize total joint costs, Smith (1956) showed that the players must be ordered with respect to weakly decreasing urgency, defined as the ratio of the individual cost parameter and the processing time. Assuming the presence of an initial order, this reordering will lead to cost savings. To analyze how to divide the maximal cost savings among the players, Curiel, Pederzoli, and Tijs (1989) introduced cooperative sequencing games. They show that sequencing games are convex and therefore have a non-empty core. This means that it is always possible to find a coalitionally stable cost savings division. A more recent review of sequencing games can be found in Curiel, Hamers, and Klijn (2002).

A common assumption underlying the definition of the coalitional values in many sequencing games is that two players of a certain coalition can only swap their positions if all players between them are also members of the coalition. Curiel, Potters, Prasad, Tijs, and Veltman (1993) argued that the resulting set of admissible reorderings for a

^{*}CentER and Department of Econometrics and Operations Research Tilburg University, P.O. Box 90153, 5000 LE Tilburg, The Netherlands

[†]Correspondence to: m.musegaas@tilburguniversity.edu

coalition is too restrictive because there may be more reorderings possible which do not hurt the interests of the players outside the coalition. Relaxed sequencing games arise by relaxing the classical assumption about the set of admissible rearrangements for coalitions in a consistent way. In Curiel et al. (1993) four different relaxed sequencing games are introduced. These relaxations are based on requirements for the players outside the coalition regarding either their position in the processing order (position unchanged/may change) or their starting time (starting time unchanged/not increased). This means that a player in a certain coalition is allowed to jump over players outside the coalition as long as the exogenously imposed requirements are satisfied. As a consequence, a player may be moved to a position earlier in the processing order when another player moves backwards. Slikker (2006) proved non-emptiness of the core for all four types of relaxed sequencing games considered in Curiel et al. (1993). In Van Velzen and Hamers (2003) two further classes of relaxed sequencing games are considered. In the first class there is a specific player allowed to switch with a player in front of him in the processing order if this player has a larger processing time, and with a player behind him in the processing order if this player has a smaller processing time. In the second class there are fixed time slots and thus only jobs with equal processing times can be switched. Van Velzen and Hamers (2003) proved that both classes of relaxed sequencing games have a non-empty core. In fact, a lot of attention has been paid to non-emptiness of the core of relaxed sequencing games. However, surprisingly enough, up to now for none of the relaxed sequencing games described above attention has been paid to finding polynomial time algorithms determining optimal processing orders for all possible coalitions.

In this paper another class of relaxed sequencing games is introduced: Step out - Step in (SoSi) sequencing games. This relaxation is intuitive from a practical point of view, because in this relaxation a member of a coalition is also allowed to step out from his position in the processing order and to step in at any position somewhere later in the processing order. In particular, each player outside the coalition will not obtain any new predecessors, possibly only fewer. In this paper we provide a direct proof for non-emptiness of the core of SoSi sequencing games. The proof provides techniques that are also applicable to prove non-emptiness of the core for other classes of relaxed sequencing games. This is illustrated with the class of Step out sequencing games. Moreover, we provide a polynomial time algorithm determining an optimal processing order for a coalition and the corresponding coalitional value. The algorithm considers the players of the coalition in an order that is the reverse of the initial order, and for every player the algorithm checks whether moving the player to a position later in the processing order is beneficial. This algorithm works in a greedy way in the sense that every player is moved to the position giving the highest cost savings at that moment. Moreover, every player is considered in the algorithm exactly once and every player is moved to another position in the processing order at most once.

The organization of this paper is as follows. Section 2 recalls basic definitions on one-machine sequencing situations and formally introduces SoSi sequencing games. In Section 3 it is shown that every SoSi sequencing game has a non-empty core. Section 4 provides a polynomial time algorithm to determine the coalitional values of a SoSi sequencing game. Finally, Section 5 considers another type of relaxed sequencing games, so-called Step out sequencing games, and it is examined whether the results of SoSi sequencing games can also be applied to these games.

2 SoSi sequencing games

A one-machine sequencing situation can be summarized by a tuple (N, σ_0, p, α) , where $N = \{1, \dots, n\}$ is the set of players, each with one job to be processed on the single machine. A processing order of the players can be described by a bijection $\sigma : N \rightarrow \{1, \dots, n\}$. More specifically, $\sigma(i) = k$ means that player i is in position k . Let $\Pi(N)$ denote the set of all such processing orders. The processing order $\sigma_0 \in \Pi(N)$ specifies the initial order. The processing time $p_i > 0$ of the job of player i is the time the machine takes to process this job. The vector $p \in \mathbb{R}_{++}^N$ summarizes the processing times. Furthermore, the costs of player i of spending t time units in the system is assumed to be determined by a linear cost function $c_i : [0, \infty) \rightarrow \mathbb{R}$ given by $c_i(t) = \alpha_i t$ with $\alpha_i > 0$. The vector $\alpha \in \mathbb{R}_{++}^N$ summarizes the coefficients of the linear cost functions. It is assumed that the machine starts processing at time $t = 0$, and also that all jobs enter the system at $t = 0$.

Let $C_i(\sigma)$ be the completion time of the job of player i with respect to processing order σ via the associated semi-active schedule, i.e., a schedule in which there is no idle time between the jobs. Hence, the completion time of player i equals

$$C_i(\sigma) = \sum_{j \in N: \sigma(j) \leq \sigma(i)} p_j.$$

A processing order is called *optimal* if the total joint costs $\sum_{i \in N} \alpha_i C_i(\sigma)$ are minimized. In Smith (1956) it is shown that in each optimal order the players are processed in non-increasing order with respect to their urgency u_i defined by $u_i = \frac{\alpha_i}{p_i}$. Moreover, with g_{ij} representing the gain made by a possible neighbor switch of i and j if player i is directly in front of player j , i.e., with

$$g_{ij} = \max\{\alpha_j p_i - \alpha_i p_j, 0\},$$

the maximal total cost savings are equal to

$$\sum_{i \in N} \alpha_i C_i(\sigma_0) - \sum_{i \in N} \alpha_i C_i(\sigma^*) = \sum_{i, j \in N: \sigma_0(i) < \sigma_0(j)} g_{ij}, \quad (1)$$

where σ^* denotes an optimal order.

A *coalitional game* is a pair (N, v) where $N = \{1, \dots, n\}$ denotes a non-empty, finite set of players and $v : 2^N \rightarrow \mathbb{R}$ assigns a monetary payoff to each coalition $S \in 2^N$, where 2^N denotes the collection of all subsets of N . The value $v(S)$ denotes the highest payoff the coalition S can jointly generate by means of optimal cooperation without help of players in $N \setminus S$. By convention, $v(\emptyset) = 0$.

To tackle the allocation problem of the maximal cost savings in a sequencing situation (N, σ_0, p, α) one can analyze an associated coalitional game (N, v) . Here N naturally corresponds to the set of players in the game and, for a coalition $S \subset N$, $v(S)$ reflects the maximal cost savings this coalition can make with respect to the initial order σ_0 . In order to determine these maximal cost savings, assumptions must be made on the possible reorderings of coalition S with respect to the initial order σ_0 .

The classical (strong) assumption is that a member of a certain coalition $S \subset N$ can only swap with another member of the coalition if all players between these two players, according to the initial order, are also members of S . This assumption results in the definition of a classical sequencing game. However, note that the resulting set

of admissible reorderings for a coalition is quite restrictive, because there may be more reorderings possible which do not hurt the interests of the players outside the coalition.

In a SoSi sequencing game the classical assumption is relaxed by additionally allowing that a member of the coalition S steps out from his position in the processing order and steps in at any position later in the processing order. Hence, a processing order σ is called *admissible* for S in a SoSi sequencing game if

- (i) $P(\sigma, i) \subset P(\sigma_0, i)$ for all $i \in N \setminus S$,
- (ii) $\sigma^{-1}(\sigma(i) + 1) \in F(\sigma_0, i)$ for all $i \in N \setminus S$,

where $P(\sigma, i) = \{j \in N \mid \sigma(j) < \sigma(i)\}$ denotes the set of predecessors of player i with respect to processing order σ and $F(\sigma, i) = \{j \in N \mid \sigma(j) > \sigma(i)\}$ denotes the set of followers. Condition (i) ensures that no player outside S obtains any new predecessors. As a result, a player who steps out from his position in the processing order, steps in at a position later, and thus not earlier, in the processing order. Note that from an optimality point of view it is clear that we can assume without loss of generality that a member of S who steps out, only steps in at a position directly behind another member of the coalition S . Therefore, condition (ii) requires that each player outside S has a direct follower who was already a follower of him with respect to σ_0 . Given an initial order σ_0 the set of admissible orders for coalition S is denoted by $\mathcal{A}(\sigma_0, S)$.

Correspondingly, the *Step out, Step in (SoSi) sequencing game* (N, v) is defined by

$$v(S) = \max_{\sigma \in \mathcal{A}(\sigma_0, S)} \sum_{i \in S} \alpha_i (C_i(\sigma_0) - C_i(\sigma))$$

for all $S \subset N$, i.e., the value of a coalition is equal to the maximal cost savings a coalition can achieve by means of admissible rearrangements. A processing order $\sigma^* \in \mathcal{A}(\sigma_0, S)$ is called *optimal* for S if

$$\sum_{i \in S} \alpha_i (C_i(\sigma_0) - C_i(\sigma^*)) = \max_{\sigma \in \mathcal{A}(\sigma_0, S)} \sum_{i \in S} \alpha_i (C_i(\sigma_0) - C_i(\sigma)).$$

Since for a coalition S the set of admissible orders in a SoSi sequencing game is a superset of the set of admissible orders in the classical sequencing game, the value of S in a SoSi sequencing game is at least its value in the classical sequencing game. For $\sigma \in \Pi(N)$, S is called *connected* with respect to σ if for all $i, j \in S$ and $k \in N$ such that $\sigma(i) < \sigma(k) < \sigma(j)$ it holds that $k \in S$. Note that for each coalition that is connected with respect to σ_0 the set of admissible orders in the SoSi sense equals the set of admissible orders in the classical sense. This means that the value of any connected coalition is the same in the SoSi sequencing game and in the classical sequencing game. Therefore, similar to (1), it readily can be concluded that the value of a coalition S that is connected with respect to σ_0 is given by

$$v(S) = \sum_{i, j \in S: \sigma_0(i) < \sigma_0(j)} g_{ij}. \quad (2)$$

Example 2.1. Consider a one-machine sequencing situation with $N = \{1, 2, 3\}$. The vector of processing times is $p = (3, 2, 1)$ and the vector of coefficients corresponding to the linear cost functions is $\alpha = (4, 6, 5)$. Assume that the initial order is $\sigma_0 = (1 \ 2 \ 3)$. It then follows that $g_{12} = 10$, $g_{13} = 11$ and $g_{23} = 4$.

Let (N, v) be the corresponding SoSi sequencing game. Table 2.1 presents the value of all coalitions. Note that the coalitional values of the game (N, v) are equal to the

coalitional values of the classical sequencing game of this one-machine sequencing situation except for the only disconnected coalition, coalition $\{1, 3\}$. For instance, for the grand coalition N it follows from (2) that

$$v(N) = \sum_{i,j \in N: \sigma_0(i) < \sigma_0(j)} g_{ij} = g_{12} + g_{13} + g_{23} = 25.$$

S	$\{1\}$	$\{2\}$	$\{3\}$	$\{1, 2\}$	$\{1, 3\}$	$\{2, 3\}$	N
$v(S)$	0	0	0	10	3	4	25

Table 2.1: The SoSi sequencing game of Example 2.1

The disconnected coalition $\{1, 3\}$ cannot save costs in the classical sequencing game because there exists no admissible order other than the initial order. However, in the SoSi sequencing game the set of admissible orders consists of two elements:

$$\mathcal{A}(\sigma_0, \{1, 3\}) = \{(1 \ 2 \ 3), (2 \ 3 \ 1)\}.$$
¹

These processing orders are illustrated in Figure 2.1. Hence, the value of coalition $\{1, 3\}$ is given by

$$v(\{1, 3\}) = \max \left\{ 0, \sum_{i \in \{1, 3\}} \alpha_i (C_i((1 \ 2 \ 3)) - C_i((2 \ 3 \ 1))) \right\} = \max\{0, -12 + 15\} = 3. \quad \triangle$$

Figure 2.1: The admissible orders for coalition $\{1, 3\}$ in Example 2.1

3 Non-emptiness of the core

In this section we show that every SoSi sequencing game has a non-empty core, i.e., it is always possible to find a stable cost savings allocation. First, we define the definition of an urgency respecting order. The definition of an urgency respecting order is used in order to develop a well-defined procedure of consecutive movements to go from an initial order to an urgency respecting order. This procedure of consecutive movements is used to define an upper bound on the value of a coalition in a SoSi sequencing game. After that, the upper bound is applied to prove non-emptiness of the core of SoSi sequencing games.

¹Processing order $(2 \ 3 \ 1)$ means that player 2 is in the first position, player 3 in the second position and player 1 in the last position.

We start with some basic definitions and notations. The core $C(v)$ of a game (N, v) is formally defined as the set of all allocations $x \in \mathbb{R}^N$ such that $\sum_{i \in N} x_i = v(N)$ (*efficiency*) and $\sum_{i \in S} x_i \geq v(S)$ for all $S \subset N$ (*stability*). For $S \in 2^N \setminus \{\emptyset\}$, $\sigma \in \Pi(N)$ and $s, t \in N$ with $\sigma(s) < \sigma(t)$, define

$$\begin{aligned} S^\sigma(s, t) &= \{i \in S \mid \sigma(s) < \sigma(i) < \sigma(t)\}, \\ \bar{S}^\sigma(s, t) &= \{i \in N \setminus S \mid \sigma(s) < \sigma(i) < \sigma(t)\}, \\ S^\sigma[s, t] &= \{i \in S \mid \sigma(s) \leq \sigma(i) \leq \sigma(t)\}, \\ \bar{S}^\sigma[s, t] &= \{i \in N \setminus S \mid \sigma(s) \leq \sigma(i) \leq \sigma(t)\}. \end{aligned}$$

The sets of players $S^\sigma[s, t]$, $\bar{S}^\sigma[s, t]$, $S^\sigma(s, t)$ and $\bar{S}^\sigma(s, t)$ are defined in a similar way.

For $\sigma \in \Pi(N)$, a connected coalition $T \subset S$ with respect to σ is called a *component* of S with respect to σ if $T \subset T' \subset S$ and T' connected with respect to σ implies that $T' = T$. Let $h(\sigma, S) \geq 1$ denote the number of components of S with respect to σ . The partition of S into components with respect to σ is denoted by $S \setminus \sigma = \{S_1^\sigma, S_2^\sigma, \dots, S_{h(\sigma, S)}^\sigma\}$. Furthermore, we define

$$\begin{aligned} \bar{S}_0^\sigma &= \{i \in N \setminus S \mid \sigma(i) < \sigma(j) \text{ for all } j \in S_1^\sigma\}, \\ \bar{S}_{h(\sigma, S)}^\sigma &= \{i \in N \setminus S \mid \sigma(i) > \sigma(j) \text{ for all } j \in S_{h(\sigma, S)}^\sigma\}, \\ \bar{S}_k^\sigma &= \{i \in N \setminus S \mid \sigma(j) < \sigma(i) < \sigma(l) \text{ for all } j \in S_k^\sigma, \text{ for all } l \in S_{k+1}^\sigma\}, \end{aligned} \tag{3}$$

for all $k \in \{1, \dots, h(\sigma, S) - 1\}$ (cf. Figure 3.1). Notice that $\bar{S}_0^\sigma$ and $\bar{S}_{h(\sigma, S)}^\sigma$ might be empty sets, but $\bar{S}_k^\sigma \neq \emptyset$ for all $k \in \{1, \dots, h(\sigma, S) - 1\}$.

Figure 3.1: Partition of the players with respect to an order σ

Note that for given $S \subset N$ it is possible that a processing order $\sigma \in \mathcal{A}(\sigma_0, S)$ contains less components for S than σ_0 , because all players of a certain component with respect to S may step out from this component and join other components. Define *modified components* $S_1^{\sigma_0, \sigma}, \dots, S_{h(\sigma_0, S)}^{\sigma_0, \sigma}$ by

$$S_k^{\sigma_0, \sigma} = \{i \in S \mid \sigma(j) < \sigma(i) < \sigma(l) \text{ for all } j \in \bar{S}_{k-1}^{\sigma_0}, \text{ for all } l \in \bar{S}_k^{\sigma_0}\},$$

for all $k \in \{1, \dots, h(\sigma_0, S)\}$. Note that $S_k^{\sigma_0, \sigma}$ might be empty for some k while

$$\bigcup_{k=1}^{h(\sigma_0, S)} S_k^{\sigma_0, \sigma} = S,$$

and

$$\bigcup_{k=1}^l S_k^{\sigma_0, \sigma} \subset \bigcup_{k=1}^l S_k^{\sigma_0},$$

for all $l \in \{1, \dots, h(\sigma_0, S)\}$.

A processing order $\sigma \in \mathcal{A}(\sigma_0, S)$ is called *urgency respecting* if

(i) (σ is *componentwise optimal*) for all $i, j \in S_k^{\sigma_0, \sigma}$ with $k \in \{1, \dots, h(\sigma_0, S)\}$:

$$\sigma(i) < \sigma(j) \Rightarrow u_i \geq u_j.$$

(ii) (σ satisfies *partial tiebreaking*) for all $i, j \in S_k^{\sigma_0}$ with $k \in \{1, \dots, h(\sigma_0, S)\}$:

$$u_i = u_j, i < j \Rightarrow \sigma(i) < \sigma(j).$$

The partial tiebreaking condition ensures that if there are two players with the same urgency in the same component of S with respect to σ_0 , then the player with the lowest index is earlier in processing order σ . Note that the partial tiebreaking condition does not imply anything about the relative order of two players with the same urgency who are in the same component of S with respect to σ but who were in different components of S with respect to σ_0 . Therefore, an urgency respecting order does not need to be unique. Clearly, there always exists an optimal order for S that is urgency respecting. The partial tiebreaking condition is required to make sure that there is a well-defined procedure of consecutive movements to go from the initial order σ_0 to the urgency respecting order σ , as is explained later in this section.

Next, define $\sigma_0^S \in \mathcal{A}(\sigma_0, S)$ to be the unique urgency respecting processing order such that

$$S_k^{\sigma_0, \sigma_0^S} = S_k^{\sigma_0},$$

for all $k \in \{1, \dots, h(\sigma_0, S)\}$.

For a processing order $\sigma \in \Pi(N)$ and $i, j \in N$, with $\sigma(i) < \sigma(j)$, we define $[i, j]\sigma$ to be the processing order that is obtained from σ by moving player i to the position directly behind player j , i.e.,

$$([i, j]\sigma)(s) = \begin{cases} \sigma(s) & \text{if } s \notin N^\sigma[i, j] \\ \sigma(s) - 1 & \text{if } s \in N^\sigma(i, j] \\ \sigma(j) & \text{if } s = i, \end{cases}$$

for every $s \in N$ (see Figure 3.2).

Figure 3.2: Illustration of $[i, j]\sigma$

In a SoSi sequencing game there are two types of operations allowed for a coalition S given the initial order σ_0 . A type I operation is a swap of adjacent pairs in the same component of S . A type II operation is a move from a player in S to the position directly behind another player of S within one of the subsequent components.

An urgency respecting processing order $\sigma \in \mathcal{A}(\sigma_0, S)$ can always be obtained with the operations described above from the initial order σ_0 via the processing order σ_0^S . In order to obtain the processing order σ_0^S from the initial order σ_0 only type I operations

are performed, while for obtaining the processing order σ from σ_0^S only type II operations need to be performed. These type II operations can be chosen in such a way that the moved player is already on the correct urgency respecting position in his new component, as demonstrated below.

Let $\sigma \in \mathcal{A}(\sigma_0, S)$ be urgency respecting. Define $R(\sigma)$ as the set of players who switch component, i.e.,

$$R(\sigma) = \{i \in S \mid i \in S_k^{\sigma_0}, i \in S_l^{\sigma_0, \sigma} \text{ with } k, l \in \{1, \dots, h(\sigma_0, S)\} \text{ such that } l > k\}. \quad (4)$$

If $\sigma \neq \sigma_0^S$, then $|R(\sigma)| \geq 1$. Next, define $r_1(\sigma) \in R(\sigma)$ such that

$$\sigma_0^S(r_1(\sigma)) \geq \sigma_0^S(r),$$

for all $r \in R(\sigma)$ and $m_1(\sigma) \in S$ with $m_1(\sigma) \notin R(\sigma)$, such that

$$\sigma(m_1(\sigma)) < \sigma(r_1(\sigma))$$

and

$$\sigma(m_1(\sigma)) \geq \sigma(j),$$

for all $j \in S$ with $j \notin R(\sigma)$ and $\sigma(j) < \sigma(r_1(\sigma))$. Note that $m_1(\sigma)$ is well-defined because $\sigma_0^S(m_1(\sigma)) > \sigma_0^S(r_1(\sigma))$ due to condition (ii) of admissibility. Defining

$$\tau^{\sigma, S, 1} = [r_1(\sigma), m_1(\sigma)]\sigma_0^S,$$

$\tau^{\sigma, S, 1}$ is an urgency respecting and admissible order for S . Intuitively, player $r_1(\sigma)$ is the first player who is moved in order to go from the order σ_0^S to the order σ . Moreover, $m_1(\sigma)$ is the player where player $r_1(\sigma)$ must be positioned behind.

For $k \in \{2, \dots, |R(\sigma)|\}$, recursively, define $r_k(\sigma) \in R(\sigma) \setminus \{r_1(\sigma), \dots, r_{k-1}(\sigma)\}$ such that

$$\sigma_0^S(r_k(\sigma)) \geq \sigma_0^S(r),$$

for all $r \in R(\sigma) \setminus \{r_1(\sigma), \dots, r_{k-1}(\sigma)\}$. Moreover, define $m_k(\sigma) \in S$ with $m_k(\sigma) \notin R(\sigma) \setminus \{r_1(\sigma), \dots, r_{k-1}(\sigma)\}$ such that

$$\sigma(m_k(\sigma)) < \sigma(r_k(\sigma))$$

and

$$\sigma(m_k(\sigma)) \geq \sigma(j),$$

for all $j \in S$ with $j \notin R(\sigma) \setminus \{r_1(\sigma), \dots, r_{k-1}(\sigma)\}$ and $\sigma(j) < \sigma(r_k(\sigma))$, and, finally, set

$$\tau^{\sigma, S, k} = [r_k(\sigma), m_k(\sigma)]\tau^{\sigma, S, k-1}. \quad (5)$$

Note that $m_k(\sigma)$ is well-defined because $\tau^{\sigma, S, k-1}(m_k(\sigma)) > \tau^{\sigma, S, k-1}(r_k(\sigma))$ due to condition (ii) of admissibility. Moreover, $\tau^{\sigma, S, k}$ is an admissible urgency respecting order for S (because σ is urgency respecting) and

$$\tau^{\sigma, S, |R(\sigma)|} = \sigma.$$

For notational convenience we define $\tau^{\sigma, S, 0}$ to be σ_0^S .

An illustration of the procedure described above can be found in the following example.

Player i	1	2	3	4	5	6	7	8	9	10
u_i	$\frac{4}{9}$	$\frac{4}{9}$	4	$1\frac{1}{2}$	$\frac{1}{3}$	$\frac{3}{8}$	$\frac{4}{9}$	$\frac{1}{5}$	$\frac{1}{4}$	1

Table 3.1: Urgencies of the players in coalition S in Example 3.1

Example 3.1. Consider a one-machine sequencing situation (N, σ_0, p, α) with

$$S = \{1, 2, \dots, 10\},$$

such that $\sigma_0(k) < \sigma_0(l)$ if and only if $k < l$, for $k, l \in S$. Moreover, assume that the components of coalition S with respect to σ_0 are given by

$$S_1^{\sigma_0} = \{1, 2, 3\}, S_2^{\sigma_0} = \{4, 5\}, S_3^{\sigma_0} = \{6\} \text{ and } S_4^{\sigma_0} = \{7, 8, 9, 10\},$$

with the urgencies of the players in S specified in Table 3.1.

In Figure 3.3 the orders σ_0 and σ_0^S (the first two processing orders) are illustrated, together with an urgency respecting order $\sigma \in \mathcal{A}(\sigma_0, S)$ (the last processing order) for which

$$S_1^{\sigma_0, \sigma} = \{3\}, S_2^{\sigma_0, \sigma} = \{1, 4\}, S_3^{\sigma_0, \sigma} = \emptyset \text{ and } S_4^{\sigma_0, \sigma} = \{2, 5, 6, 7, 8, 9, 10\}.$$

Hence, from (4) it follows that

$$R(\sigma) = \{1, 2, 5, 6\},$$

and

$$r_1(\sigma) = 6, r_2(\sigma) = 5, r_3(\sigma) = 2 \text{ and } r_4(\sigma) = 1.$$

Moreover,

$$m_1(\sigma) = 7, m_2(\sigma) = 6, m_3(\sigma) = 7 \text{ and } m_4(\sigma) = 4.$$

The orders $\tau^{\sigma, S, 1}$, $\tau^{\sigma, S, 2}$, $\tau^{\sigma, S, 3}$ and $\tau^{\sigma, S, 4}$ are depicted in Figure 3.3 as well. Since $\tau^{\sigma, S, 4} = \sigma$ we find that

$$\sigma = [1, 4][2, 7][5, 6][6, 7]\sigma_0^S. \quad \triangle$$

Lemma 3.1 below is used in order to show non-emptiness of the core of SoSi sequencing games (see Theorem 3.2). Lemma 3.1 provides an upper bound on the value of a coalition in a SoSi sequencing game in terms of the gains made by possible neighbor switches. Note that this upper bound is tight if the coalition is connected. The proof of Theorem 3.2 provides techniques that are also applicable to prove non-emptiness of the core for a more general class of relaxed sequencing games. Namely, the proof applies to all sequencing games in which the value of every coalition satisfies (6). An example of another type relaxed sequencing game that belongs to this general class is given in Section 5.

Lemma 3.1. *Let (N, σ_0, p, α) be a one-machine sequencing situation and let (N, v) be the corresponding SoSi sequencing game. Then,*

$$v(S) \leq \sum_{i, j \in S: \sigma_0(i) < \sigma_0(j)} g_{ij}, \quad (6)$$

for all $S \subset N$.

Figure 3.3: The processing orders corresponding to Example 3.1

Proof. Let $S \subset N$ and let σ^* be an urgency respecting optimal order for coalition S . Because of (2) we can assume without loss of generality that S is not connected. We define a special processing order θ_0 such that the coalition S is connected with respect to θ_0 . Thereafter, all type I and type II operations that are used to obtain σ^* from σ_0 via σ_0^S are performed on the processing order θ_0 .

Denote by Q the set of players outside S and positioned between two components of S according to σ_0 , i.e.,

$$Q = \bigcup_{k=1}^{h(\sigma_0, S)-1} \bar{S}_k^{\sigma_0}.$$

Consider a one-machine sequencing situation (N, θ_0, p, α) with N, p and α as defined above and θ_0 an initial order such that

1. $\theta_0(i) = \sigma_0(i)$ for all $i \in N \setminus (S \cup Q)$,
2. $\sigma_0(i) < \sigma_0(j) \Rightarrow \theta_0(i) < \theta_0(j)$ for all $i, j \in S$,
3. $\min\{\theta_0(i) \mid i \in S\} = \min\{\sigma_0(i) \mid i \in S\}$,
4. $\max\{\theta_0(i) \mid i \in S\} = \min\{\sigma_0(i) \mid i \in S\} + |S| - 1$.

Hence, the order θ_0 is derived from σ_0 in such a way that

- the position of the players outside $S \cup Q$ has not been changed,
- the relative order between the players in S remains the same,
- the players in S are moved forward as far as possible.

As a consequence, all players in Q are positioned in an arbitrary way between $S_{h(\sigma_0, S)}^{\sigma_0}$ and $\bar{S}_{h(\sigma_0, S)}^{\sigma_0}$ according to θ_0 . In particular, S is a connected coalition with respect to θ_0 (cf. Figure 3.4).

Denote processing order $\hat{\theta}_0 \in \mathcal{A}(\theta_0, S)$ such that

$\bar{S}_0^{\sigma_0}$	$S_1^{\sigma_0}$	$S_2^{\sigma_0}$	$S_3^{\sigma_0}$	$\dots$	$S_{h(\sigma_0, S)}^{\sigma_0}$	Q	$\bar{S}_{h(\sigma_0, S)}^{\sigma_0}$
------------------------	------------------	------------------	------------------	---------	---------------------------------	-----	---------------------------------------

Figure 3.4: Illustration of the order θ_0

1. $\hat{\theta}_0(i) = \theta_0(i)$ for all $i \notin S$,
2. $\sigma_0^S(i) < \sigma_0^S(j) \Rightarrow \hat{\theta}_0(i) < \hat{\theta}_0(j)$ for all $i, j \in S$.

Hence, the order $\hat{\theta}_0$ is obtained from θ_0 in such a way that

- the position of the players outside S has not been changed,
- such that the relative order between the players in S is the same as their relative order in σ_0^S .

Obviously $\hat{\theta}_0$ can be obtained from θ_0 in exactly the same way as σ_0^S from σ_0 using the same operations of type I, conducted in the same order. Note that each operation results in the same cost difference. Therefore,

$$\sum_{i \in S} \alpha_i (C_i(\sigma_0) - C_i(\sigma_0^S)) = \sum_{i \in S} \alpha_i (C_i(\theta_0) - C_i(\hat{\theta}_0)). \quad (7)$$

Observe that if the order σ_0 is already urgency respecting, i.e., if $\sigma_0^S = \sigma_0$, then also $\hat{\theta}_0 = \theta_0$ and thus (7) still holds.

Next, let $R = R(\sigma^*)$, $r_k = r_k(\sigma^*)$ and $m_k = m_k(\sigma^*)$ for all $k \in \{1, \dots, |R|\}$ as defined in (4) such that

$$\sigma^* = [r_{|R|}, m_{|R|}] \dots [r_2, m_2][r_1, m_1]\sigma_0^S.$$

Define the processing order θ^* by

$$\theta^* = [r_{|R|}, m_{|R|}] \dots [r_2, m_2][r_1, m_1]\hat{\theta}_0.$$

Remember, if $|R| = 0$, then $\sigma^* = \sigma_0^S$ and thus also $\theta^* = \hat{\theta}_0$. The processing order θ^* is obtained from $\hat{\theta}_0$ in the same way as σ^* is obtained from σ_0^S using the same type II operations, conducted in the same order. Note that these operations are indeed valid due to the definition of $\hat{\theta}_0$. Obviously, θ^* is an admissible order for S in the sequencing situation (N, θ_0, p, α) . Abbreviate $\tau^{\sigma^*, S, k}$ for $k \in \{0, 1, \dots, |R|\}$ as defined in (5) by $\hat{\sigma}_k$. Moreover, set

$$\hat{\theta}_k = [r_k, m_k] \dots [r_2, m_2][r_1, m_1]\hat{\theta}_0,$$

for all $k \in \{0, 1, \dots, |R|\}$. Notice that $\hat{\theta}_{|R|} = \theta^*$.

Let $k \in \{1, \dots, |R|\}$ and consider the operation $[r_k, m_k]$ performed on $\hat{\sigma}_{k-1}$ and $\hat{\theta}_{k-1}$.

Then,

$$\begin{aligned}
\sum_{i \in S} \alpha_i (C_i(\hat{\sigma}_{k-1}) - C_i(\hat{\sigma}_k)) &= \left(\sum_{i \in S^{\hat{\sigma}_{k-1}}(r_k, m_k]} \alpha_i \right) p_{r_k} - \alpha_{r_k} \left(\sum_{i \in N^{\hat{\sigma}_{k-1}}(r_k, m_k]} p_i \right) \\
&\leq \left(\sum_{i \in S^{\hat{\sigma}_{k-1}}(r_k, m_k]} \alpha_i \right) p_{r_k} - \alpha_{r_k} \left(\sum_{i \in S^{\hat{\sigma}_{k-1}}(r_k, m_k]} p_i \right) \\
&= \left(\sum_{i \in S^{\hat{\theta}_{k-1}}(r_k, m_k]} \alpha_i \right) p_{r_k} - \alpha_{r_k} \left(\sum_{i \in S^{\hat{\theta}_{k-1}}(r_k, m_k]} p_i \right) \\
&= \sum_{i \in S} \alpha_i (C_i(\hat{\theta}_{k-1}) - C_i(\hat{\theta}_k)).
\end{aligned}$$

Hence,

$$\begin{aligned}
v(S) &= \sum_{i \in S} \alpha_i (C_i(\sigma_0) - C_i(\sigma^*)) \\
&= \sum_{i \in S} \alpha_i (C_i(\sigma_0) - C_i(\sigma_0^S)) + \sum_{k=1}^{|R|} \sum_{i \in S} \alpha_i (C_i(\hat{\sigma}_{k-1}) - C_i(\hat{\sigma}_k)) \\
&\leq \sum_{i \in S} \alpha_i (C_i(\theta_0) - C_i(\hat{\theta}_0)) + \sum_{k=1}^{|R|} \sum_{i \in S} \alpha_i (C_i(\hat{\theta}_{k-1}) - C_i(\hat{\theta}_k)) \\
&= \sum_{i \in S} \alpha_i (C_i(\theta_0) - C_i(\theta^*)) \\
&\leq \max_{\theta \in \mathcal{A}(\theta_0, S)} \sum_{i \in S} \alpha_i (C_i(\theta_0) - C_i(\theta)) \\
&= \sum_{i, j \in S: \theta_0(i) < \theta_0(j)} g_{ij} \\
&= \sum_{i, j \in S: \sigma_0(i) < \sigma_0(j)} g_{ij},
\end{aligned}$$

also if $|R| = 0$. In the above derivation, the last two equalities follow from

- the fact that coalition S is connected with respect to θ_0 ,
- the fact that the relative order of the players in S with respect to θ_0 is the same as the relative order of the players in S with respect to σ_0 . $\square$

Now we are ready to prove non-emptiness of the core of every SoSi sequencing game.

Theorem 3.2. *Let (N, σ_0, p, α) be a one-machine sequencing situation. Then, the corresponding SoSi sequencing game (N, v) has a non-empty core.*

Proof. Let $\pi \in \Pi(N)$ be a connected order with respect to σ_0 , i.e., π is such that the set $\{j \in N \mid \pi(j) \leq \pi(i)\}$ is connected with respect to σ_0 for all $i \in N$. We show that the corresponding marginal vector $m^\pi(v)$ belongs to the core $C(v)$.

Let $i \in N$ and denote $T_i = \{j \in N \mid \pi(j) < \pi(i)\}$. Because π is connected it follows that either $T_i \subset P(\sigma_0, i)$ or $T_i \subset F(\sigma_0, i)$ and therefore we have

$$\begin{aligned} m_i^\pi(v) &= v(T_i \cup \{i\}) - v(T_i) \\ &= \sum_{j,k \in T_i \cup \{i\}: \sigma_0(j) < \sigma_0(k)} g_{jk} - \sum_{j,k \in T_i: \sigma_0(j) < \sigma_0(k)} g_{jk} \\ &= \begin{cases} \sum_{j \in T_i} g_{ji} & \text{if } T_i \subset P(\sigma_0, i) \\ \sum_{j \in T_i} g_{ij} & \text{if } T_i \subset F(\sigma_0, i), \end{cases} \end{aligned}$$

where the second equality follows from the fact that the coalitions T_i and $T_i \cup \{i\}$ are connected with respect to σ_0 .

Then, for $S \subset N$ it holds that

$$\begin{aligned} \sum_{i \in S} m_i^\pi(v) &= \sum_{\substack{i \in S: \\ T_i \subset P(\sigma_0, i)}} \sum_{j \in T_i} g_{ji} + \sum_{\substack{i \in S: \\ T_i \subset F(\sigma_0, i)}} \sum_{j \in T_i} g_{ij} \\ &\geq \sum_{\substack{i \in S: \\ T_i \subset P(\sigma_0, i)}} \sum_{\substack{j \in T_i: \\ j \in P(\sigma_0, i) \cap S}} g_{ji} + \sum_{\substack{i \in S: \\ T_i \subset F(\sigma_0, i)}} \sum_{\substack{j \in T_i: \\ j \in F(\sigma_0, i) \cap S}} g_{ij} \\ &= \sum_{\substack{i \in S: \\ T_i \subset P(\sigma_0, i)}} \sum_{\substack{j \in T_i: \\ j \in P(\sigma_0, i) \cap S}} g_{ji} + \sum_{\substack{i \in S: \\ T_i \subset F(\sigma_0, i)}} \sum_{\substack{j \in T_i: \\ j \in P(\sigma_0, i) \cap S}} g_{ji} \\ &\quad + \sum_{\substack{i \in S: \\ T_i \subset P(\sigma_0, i)}} \sum_{\substack{j \in T_i: \\ j \in F(\sigma_0, i) \cap S}} g_{ij} + \sum_{\substack{i \in S: \\ T_i \subset F(\sigma_0, i)}} \sum_{\substack{j \in T_i: \\ j \in F(\sigma_0, i) \cap S}} g_{ij} \\ &= \sum_{i \in S} \sum_{\substack{j \in T_i: \\ j \in P(\sigma_0, i) \cap S}} g_{ji} + \sum_{i \in S} \sum_{\substack{j \in T_i: \\ j \in F(\sigma_0, i) \cap S}} g_{ij}, \\ &= \sum_{j \in S} \sum_{\substack{i \in T_j: \\ i \in F(\sigma_0, j) \cap S}} g_{ji} + \sum_{i \in S} \sum_{\substack{j \in T_i: \\ j \in F(\sigma_0, i) \cap S}} g_{ij} \\ &= \sum_{i \in S} \sum_{\substack{i \in T_j: \\ j \in F(\sigma_0, i) \cap S}} g_{ij} + \sum_{i \in S} \sum_{\substack{j \in T_i: \\ j \in F(\sigma_0, i) \cap S}} g_{ij} \\ &= \sum_{i \in S} \sum_{\substack{j \notin T_i: \\ j \in F(\sigma_0, i) \cap S}} g_{ij} + \sum_{i \in S} \sum_{\substack{j \in T_i: \\ j \in F(\sigma_0, i) \cap S}} g_{ij} \\ &= \sum_{i \in S} \sum_{j \in F(\sigma_0, i) \cap S} g_{ij} \\ &= \sum_{i, j \in S: \sigma_0(i) < \sigma_0(j)} g_{ij} \\ &\geq v(S). \end{aligned}$$

In the above derivation, the equations follow from

- the fact that $P(\sigma_0, i) \cap F(\sigma_0, i) = \emptyset$ for all $i \in N$ (third equality),
- interchanging the summations of the first term (fifth equality),

- interchanging the indices of the summations of the first term (sixth equality),
- the fact that $i \in T_j$ if and only if $j \notin T_i$ for all $i, j \in N$ with $i \neq j$ (seventh equality),
- Lemma 3.1 (last inequality).

Note that if $S = N$ the inequalities become equalities. This proves that $m^\pi(v) \in C(v)$. $\square$

4 On finding the coalitional values

This section provides a polynomial time algorithm determining an optimal order for every possible coalition and, consequently, the coalitional values. First, some new notions as composed costs per time unit, composed processing times and composed urgencies are introduced. Using these notions we can exclude some admissible orders from being optimal.

The *composed costs per time unit* α_S and the *composed processing time* p_S for a coalition $S \in 2^N \setminus \{\emptyset\}$ are defined by

$$\alpha_S = \sum_{i \in S} \alpha_i$$

and

$$p_S = \sum_{i \in S} p_i,$$

respectively. The *composed urgency* u_S of a non-empty coalition $S \in 2^N \setminus \{\emptyset\}$ is defined by

$$u_S = \frac{\alpha_S}{p_S}.$$

Next we explain how the concept of composed urgency helps to decide which of two related processing orders is less costly for a certain coalition. For the moment we do not worry about admissibility of these orders. Consider a non-empty coalition S and a processing order $\sigma \in \Pi(N)$. First take $i \in S$ and $j \in N$ such that $\sigma(i) < \sigma(j)$. Let $\hat{\sigma} \in \Pi(N)$ be the processing order obtained from σ by moving player i to the position directly behind player j . Then the difference between the costs for coalition S with respect to the processing order σ and the processing order $\hat{\sigma}$ can be calculated as follows:

$$\begin{aligned} & \sum_{s \in S} \alpha_s C_s(\sigma) - \sum_{s \in S} \alpha_s C_s(\hat{\sigma}) \\ &= \sum_{s \in S^\sigma(i,j]} \alpha_s p_i - \alpha_i \sum_{s \in N^\sigma(i,j]} p_s \\ &= \alpha_{S^\sigma(i,j]} p_i - \alpha_i p_{N^\sigma(i,j]}. \end{aligned}$$

Similarly, take $i \in S$ and $j \in N$ such that $\sigma(i) > \sigma(j)$ and let $\tilde{\sigma} \in \Pi(N)$ be the processing order obtained from σ by moving player i to the position directly in front of player j . Then the difference between the costs for coalition S with respect to processing order σ and processing order $\tilde{\sigma}$ is equal to

$$\alpha_i p_{N^\sigma(j,i)} - \alpha_{S^\sigma(j,i)} p_i.$$

Hence, with $\delta_i^\sigma(j, S)$ representing the cost difference for coalition S made by moving player i directly behind player j in case $\sigma(i) < \sigma(j)$ and the cost difference made by moving player i directly in front of player j in case $\sigma(i) > \sigma(j)$, we have

$$\delta_i^\sigma(j, S) = \begin{cases} \alpha_{S^{\sigma(i,j)}} p_i - \alpha_i p_{N^{\sigma(i,j)}} & \text{if } \sigma(i) < \sigma(j) \\ \alpha_i p_{N^{\sigma[j,i]}} - \alpha_{S^{\sigma[j,i]}} p_i & \text{if } \sigma(i) > \sigma(j), \end{cases} \quad (8)$$

for all $i \in S$ and $j \in N$.

The above cost differences have an additive structure in the following sense. Take $i \in S$ and $j, k \in N$ such that $\sigma(i) < \sigma(j) < \sigma(k)$. The cost difference $\delta_i^\sigma(k, S)$ can be split up in two parts by first moving player i to the position directly behind player j and thereafter moving player i to the position directly behind player k . Hence, we can write

$$\delta_i^\sigma(k, S) = \delta_i^\sigma(j, S) + \delta_i^{\hat{\sigma}}(k, S),$$

where $\hat{\sigma} \in \Pi(N)$ is the processing order obtained from σ by moving player i to the position directly behind player j .

Using the above notation of a cost difference it is easily checked whether having a certain player $i \in S$ in a later position (behind player j) in the processing order is beneficial for S or not because

$$\begin{aligned} \delta_i^\sigma(j, S) > 0 &\Leftrightarrow \frac{\alpha_i}{p_i} < \frac{\alpha_{S^{\sigma(i,j)}}}{p_{N^{\sigma(i,j)}}}, \\ \delta_i^\sigma(j, S) = 0 &\Leftrightarrow \frac{\alpha_i}{p_i} = \frac{\alpha_{S^{\sigma(i,j)}}}{p_{N^{\sigma(i,j)}}}, \\ \delta_i^\sigma(j, S) < 0 &\Leftrightarrow \frac{\alpha_i}{p_i} > \frac{\alpha_{S^{\sigma(i,j)}}}{p_{N^{\sigma(i,j)}}}, \end{aligned}$$

with $\sigma(i) < \sigma(j)$. Similarly, it is also easily checked whether having a certain player $i \in S$ on a position earlier (in front of player j) in the processing order is beneficial for S or not because

$$\begin{aligned} \delta_i^\sigma(j, S) > 0 &\Leftrightarrow \frac{\alpha_i}{p_i} > \frac{\alpha_{S^{\sigma[j,i]}}}{p_{N^{\sigma[j,i]}}}, \\ \delta_i^\sigma(j, S) = 0 &\Leftrightarrow \frac{\alpha_i}{p_i} = \frac{\alpha_{S^{\sigma[j,i]}}}{p_{N^{\sigma[j,i]}}}, \\ \delta_i^\sigma(j, S) < 0 &\Leftrightarrow \frac{\alpha_i}{p_i} < \frac{\alpha_{S^{\sigma[j,i]}}}{p_{N^{\sigma[j,i]}}}, \end{aligned}$$

with $\sigma(i) > \sigma(j)$.

Using these criteria one can exclude some admissible orders from being optimal, as illustrated in the following lemma. To be specific, Lemma 4.1 states that if there is a player $t \in S$ positioned according to $\sigma \in \mathcal{A}(\sigma_0, S)$ behind a player $s \in S$ with lower urgency and it is admissible that these players swap their position then σ cannot be an optimal order, i.e., σ can be improved.

Lemma 4.1. *Let (N, σ_0, p, α) be a one-machine sequencing situation, let $S \in 2^N \setminus \{\emptyset\}$ and let $\sigma \in \mathcal{A}(\sigma_0, S)$. Let $s, t \in S$ such that $\sigma(s) < \sigma(t)$, $u_s < u_t$ and $\sigma_0(t) < \sigma_0(i)$ for all $i \in \bar{S}^\sigma(s, t)$. Then, σ is not an optimal order for coalition S .*

Proof. If $N^\sigma(s, t) = \emptyset$, then players s and t are neighbors with respect to σ , so interchanging the positions of players s and t results in cost savings of $\alpha_t p_s - \alpha_s p_t > 0$ because $u_s < u_t$. Hence, σ cannot be an optimal order for coalition S .

Hence, $N^\sigma(s, t) \neq \emptyset$ and denote the first player of $N^\sigma(s, t)$ with respect to σ by k . We distinguish between three different cases by considering $\frac{\alpha_t}{p_t}$ and $\frac{\alpha_{S^\sigma(s, t)}}{p_{N^\sigma(s, t)}}$, and show that for every case we can find an admissible order for which the total costs for coalition S is less than the total costs with respect to processing order σ .

Case 1: $\frac{\alpha_t}{p_t} > \frac{\alpha_{S^\sigma(s, t)}}{p_{N^\sigma(s, t)}}$. Consider the processing order σ_1 obtained from σ by moving player t to the position directly behind player s (which is the position directly in front of player k , cf. Figure 4.1). Notice that the processing order σ_1 fulfills condition (i) of admissibility because all players in $\bar{S}^\sigma(s, t)$ are positioned behind player t with respect to σ_0 while condition (ii) is satisfied trivially. According to (8) the resulting cost difference for coalition S with respect to σ and σ_1 is

$$\begin{aligned}\delta_t^\sigma(k, S) &= \alpha_t p_{N^\sigma[k, t]} - \alpha_{S^\sigma[k, t]} p_t \\ &= \alpha_t p_{N^\sigma(s, t)} - \alpha_{S^\sigma(s, t)} p_t > 0,\end{aligned}$$

and thus σ is not optimal for coalition S .

Case 2: $\frac{\alpha_t}{p_t} = \frac{\alpha_{S^\sigma(s, t)}}{p_{N^\sigma(s, t)}}$. Consider again the admissible processing order σ_1 defined in Case 1. In this case the resulting cost difference for coalition S with respect to σ and σ_1 is

$$\begin{aligned}\delta_t^\sigma(k, S) &= \alpha_t p_{N^\sigma[k, t]} - \alpha_{S^\sigma[k, t]} p_t \\ &= \alpha_t p_{N^\sigma(s, t)} - \alpha_{S^\sigma(s, t)} p_t = 0.\end{aligned}$$

Consider processing order σ_2 obtained from σ_1 by interchanging the positions of players s and t (cf. Figure 4.1). This order is also admissible and the resulting cost difference for coalition S with respect to σ_1 and σ_2 is $\alpha_t p_s - \alpha_s p_t > 0$ since $u_s < u_t$. Therefore,

$$\begin{aligned}\sum_{i \in S} \alpha_i (C_i(\sigma) - C_i(\sigma_2)) &= \sum_{i \in S} \alpha_i (C_i(\sigma) - C_i(\sigma_1)) + \sum_{i \in S} \alpha_i (C_i(\sigma_1) - C_i(\sigma_2)) \\ &= 0 + \alpha_t p_s - \alpha_s p_t > 0,\end{aligned}$$

and thus σ is not optimal for coalition S .

Case 3: $\frac{\alpha_t}{p_t} < \frac{\alpha_{S^\sigma(s, t)}}{p_{N^\sigma(s, t)}}$. In this case $S^\sigma(s, t) \neq \emptyset$. Let l be the last player in S between s and t . Consider the processing order σ_3 obtained from σ by moving player s to the position directly behind player l (cf. Figure 4.1). Then, σ_3 is admissible for S and by (8) the resulting cost difference for coalition S with respect to σ and σ_3 is

$$\begin{aligned}\delta_s^\sigma(l, S) &= \alpha_{S^\sigma(s, l]} p_s - \alpha_s p_{N^\sigma(s, l]} \\ &\geq \alpha_{S^\sigma(s, t)} p_s - \alpha_s p_{N^\sigma(s, t)} > 0,\end{aligned}$$

where the last inequality follows from $u_s < u_t$ and thus $\frac{\alpha_s}{p_s} < \frac{\alpha_{S^\sigma(s, t)}}{p_{N^\sigma(s, t)}}$. Hence, σ is not an optimal order for coalition S . $\square$

Given a coalition S , Algorithm 1 below determines an urgency respecting optimal order and the corresponding cost savings with respect to σ_0 . The algorithm starts in Step

Figure 4.1: The various processing orders used in the proof of Lemma 4.1

0 with reordering the players within components of S with respect to σ_0 . This is done by setting the current processing order σ equal to the urgency respecting processing order σ_0^S . The cost savings resulting from these rearrangements are equal to the maximum gain made by all possible neighbor switches within the components as given by

$$\sum_{j=1}^{h(\sigma_0, S)} \sum_{s, t \in S_j^{\sigma_0} : \sigma_0(s) < \sigma_0(t)} g_{st}.$$

Hence, $v(S)$ can be initialized by this value, as is done in Step 1.

Subsequently, the players are considered in reverse order with respect to σ_0^S . Note that it is not admissible for a player from the last component with respect to σ_0 to step out from his component. Hence, the algorithm does not consider the players in the last component and the first player to be considered is the last player of the penultimate component. This initialization is done in Step 1 where k represents the current modified component and i represents the current player.

In Step 2 it is checked whether moving the current player i to a later modified component is beneficial. When one wants to move player i from the modified component k to a later modified component l , the position of this player in modified component l is fixed because the processing order must remain urgency respecting. This is guaranteed by considering the unique option to move the player directly behind the last player in modified component l with strictly higher urgency than player i . In this way, the partial tiebreaking condition is satisfied if later in the course of algorithm's run another player is moved to the same modified component. So, player i is never moved to a subsequent component having no players with higher urgency than player i . Let $A \subset \{k+1, \dots, h(\sigma_0, S)\}$ correspond to all later modified components that contain at least one player with higher urgency than player i . If $A = \emptyset$, then moving player i cannot be beneficial and the processing order is not adapted. With $l \in A$, define t_l as the last player in $S_l^{\sigma_0, \sigma}$ with strictly higher urgency than player i . According to (8), for every $l \in A$ the resulting cost savings due to moving player i to $S_l^{\sigma_0, \sigma}$ are equal to $\delta_i^\sigma(t_l, S)$. If these costs savings are nonpositive for all $l \in A$, then moving player i is not beneficial for S and the processing order is not adapted. On the other hand, if $\delta_i^\sigma(t_l, S)$ is positive for some $l \in A$, then player i is moved to the modified component giving the highest cost savings. Moreover, when there is a tie then the component with the smallest index is chosen (such that the partial tiebreaking condition is still satisfied at a later stage if another player joins the same component). With l^* representing the index of the new modified component of player i , processing

order σ is adapted by moving player i to the position directly behind player t_{l^*} and the current value of the coalition $v(S)$ is increased by $\delta_i^\sigma(t_{l^*}, S)$.

In Step 3 the settings of k and i are updated. If player i was not the first player of his component, then his old predecessor is the new considered player while going back to Step 2. Otherwise, the last player of the component preceding the old component of player i is considered while going back to Step 2. Moreover, if player i was the first player of coalition S with respect to processing order σ_0^S , then all players have been considered and the algorithm stops. Note that the algorithm is polynomial and thus terminates in a finite number of steps.

Algorithm 1

Input: a one-machine sequencing situation (N, σ_0, p, α) , a coalition $S \in 2^N \setminus \{\emptyset\}$

Output: an urgency respecting optimal order $\sigma \in \mathcal{A}(\sigma_0, S)$, the value $v(S)$

Step 0 (Preprocessing Step)

$\sigma := \sigma_0^S$ ▷ Order the players within the components

Step 1 (Initialization)

$v(S) := \sum_{j=1}^{h(\sigma_0, S)} \sum_{s, t \in S_j^{\sigma_0}: \sigma_0(s) < \sigma_0(t)} g_{st}$ ▷ Initialize $v(S)$ with the cost savings made in Step 0

$k := h(\sigma_0, S) - 1$ ▷ Begin with the penultimate component

$i := \operatorname{argmax}_{j \in S_k^{\sigma_0}} \sigma(j)$ ▷ Begin with the last player of this component

Step 2 (Improve Solution)

$A := \{l \in \{k+1, \dots, h(\sigma_0, S)\} \mid \exists j \in S_l^{\sigma_0, \sigma} : u_j > u_i\}$

$t_l := \operatorname{argmax}\{\sigma(j) \mid u_j > u_i, j \in S_l^{\sigma_0, \sigma}\}$ for all $l \in A$

if $A \neq \emptyset$ **and** $\max_{l \in A} \delta_i^\sigma(t_l, S) > 0$ **then** ▷ Check whether improvement is possible

$l^* := \min \{l \in A \mid \delta_i^\sigma(t_l, S) = \max_{m \in A} \delta_i^\sigma(t_m, S)\}$

$\sigma_{\text{old}} := \sigma$

$\sigma := [i, t_{l^*}] \sigma$ ▷ Revise the processing order

$v(S) := v(S) + \delta_i^\sigma(t_{l^*}, S)$ ▷ Revise the value of the coalition

end if

Step 3 (Update Settings)

if $\sigma_{\text{old}}^{-1}(\sigma_{\text{old}}(i) - 1) \in S$ **then** ▷ Was player i the first player of his component?

$i := \sigma_{\text{old}}^{-1}(\sigma_{\text{old}}(i) - 1)$ ▷ If not, consider the old predecessor of i

Go to Step 2

else

$k := k - 1$ ▷ If yes, go to the previous component

if $k > 0$ **then**

$i := \operatorname{argmax}\{\sigma(j) \mid j \in S_k^{\sigma_0, \sigma}\}$ ▷ Take the last player of this component

Go to Step 2

else

STOP ▷ All players are considered

end if

end if

Let σ_S^{alg} denote the urgency respecting processing order obtained from Algorithm 1 with respect to coalition S . Note that $R(\sigma_S^{\text{alg}}) \subset S$, as defined in (4) in Section 3, exactly

consists of those players for whom an improvement was found in Step 2. Moreover, the players $r_1(\sigma_S^{\text{alg}}), \dots, r_{|R(\sigma_S^{\text{alg}})|}(\sigma_S^{\text{alg}})$ are exactly the players considered by the algorithm, conducted in the same order. Furthermore, $m_k(\sigma_S^{\text{alg}})$ with $k \in \{1, \dots, |R(\sigma_S^{\text{alg}})|\}$ corresponds to the player where player $r_k(\sigma_S^{\text{alg}})$ is positioned behind (the corresponding t_{l^*}) according to the algorithm. Note that processing order $\tau^{\sigma_S^{\text{alg}}, S, k-1}$ is the processing order obtained by algorithm when player $r_k(\sigma_S^{\text{alg}})$ is considered and therefore the total cost savings obtained by the algorithm are equal to

$$\sum_{j=1}^{h(\sigma_0, S)} \sum_{s, t \in S_j^{\sigma_0} : \sigma_0(s) < \sigma_0(t)} g_{st} + \sum_{k=1}^{|R(\sigma_S^{\text{alg}})|} \delta_{r_k(\sigma_S^{\text{alg}})}^{\tau^{\sigma_S^{\text{alg}}, S, k-1}}(m_k(\sigma_S^{\text{alg}}), S).$$

In the example below Algorithm 1 is explained step by step.

Example 4.1. Consider a one-machine sequencing situation (N, σ_0, p, α) with

$$S = \{1, \dots, 10\}.$$

In Figure 4.2 an illustration can be found of initial order σ_0 together with all relevant data on the cost coefficients and processing times (the numbers above and below the players, respectively). The completion times of the players with respect to this initial order are also indicated in the figure (bottom line in bold).

	4	3	4		6	3		3		4	2	2	7		α_i	
$\bar{S}_0^{\sigma_0}$	1	2	3	$\bar{S}_1^{\sigma_0}$	4	5	$\bar{S}_2^{\sigma_0}$	6	$\bar{S}_3^{\sigma_0}$	7	8	9	10	$\bar{S}_4^{\sigma_0}$	σ_0	
1	9	6	1	5	4	9	5	8	3	9	10	8	7	2	p_i	
	1	10	16	17	22	26	35	40	48	51	60	70	78	85	87	$C_i(\sigma_0)$

Figure 4.2: Initial order σ_0 in Example 4.1

In Step 0 of the algorithm σ is set to processing order σ_0^S , see Figure 4.3. Note that $h(\sigma_0, S) = 4$. The resulting gains yielded by these switches within the components are

$$\sum_{j=1}^4 \sum_{s, t \in S_j^{\sigma_0} : \sigma_0(s) < \sigma_0(t)} g_{st} = 193,$$

We initialize $v(S) := 193$. In Step 1, the component to be considered first is the penultimate component, i.e., $k := 3$, and the first player to be considered is the last player of this component, which is player 6, so $i := 6$.

	4	3	4		6	3		3		7	4	2	2		α_i	
$\bar{S}_0^{\sigma_0}$	3	2	1	$\bar{S}_1^{\sigma_0}$	4	5	$\bar{S}_2^{\sigma_0}$	6	$\bar{S}_3^{\sigma_0}$	10	7	9	8	$\bar{S}_4^{\sigma_0}$	$\sigma := \sigma_0^S$	
1	1	6	9	5	4	9	5	8	3	7	9	8	10	2	p_i	
	1	2	8	17	22	26	35	40	48	51	58	67	75	85	87	$C_i(\sigma_0)$

Figure 4.3: The processing order σ after Step 0 in Example 4.1

$k = 3, i = 6$: Since there exists a player in $S_4^{\sigma_0, \sigma}$ who is more urgent than player 6, for example player 10, we have $A := \{4\}$. This means that it might be beneficial to move

player 6 to component $S_4^{\sigma_0, \sigma}$. Since player 7 is the last player in component $S_4^{\sigma_0, \sigma}$ with higher urgency than player 6, player 6 should be moved to the position directly behind player 7 if he is moved to component $S_4^{\sigma_0, \sigma}$, so $t_4 := 7$. Moving player 6 to this component results in a cost difference of

$$\begin{aligned}\delta_6^\sigma(7, S) &= \alpha_{S^\sigma(6,7]}p_6 - \alpha_6 p_{N^\sigma(6,7]} \\ &= (\alpha_{10} + \alpha_7)p_6 - \alpha_6(p_{\bar{S}_3^{\sigma_0}} + p_{10} + p_7) \\ &= 11 \cdot 8 - 3 \cdot 19 = 31.\end{aligned}$$

Since $\delta_6^\sigma(7, S) > 0$, it is beneficial to move player 6 and $l^* = 4$. Hence, we update the processing order σ by moving player 6 to the position directly behind player 7 and we set $v(S) := 193 + \delta_6^\sigma(7, S) = 224$. The updated processing order σ can be found in Figure 4.4.

$\bar{S}_0^{\sigma_0}$	3	2	1	$\bar{S}_1^{\sigma_0}$	4	5	$\bar{S}_2^{\sigma_0}$	$\bar{S}_3^{\sigma_0}$	10	7	6	9	8	$\bar{S}_4^{\sigma_0}$
------------------------	---	---	---	------------------------	---	---	------------------------	------------------------	----	---	---	---	---	------------------------

Figure 4.4: The processing order σ after player 6 is considered in Example 4.1

In Step 3 we have to update k and i . Since player 6 was the only player in his component, and thus his previous predecessor is not a member of coalition S , we consider next the component that was in front of player 6 and thus $k := 2$. The considered player is the last player of this component, i.e., $i := 5$.

$k = 2, i = 5$: Since $S_3^{\sigma_0, \sigma} = \emptyset$ we know $3 \notin A$. Moreover, as $S_4^{\sigma_0, \sigma}$ does contain a player who is more urgent than player 5, we have $A := \{4\}$. According to the given urgencies, player 5 should be moved to the position directly behind player 6 if he is moved to component $S_4^{\sigma_0, \sigma}$ ($t_4 := 6$). The resulting cost savings are

$$\begin{aligned}\delta_5^\sigma(6, S) &= \alpha_{S^\sigma(5,6]}p_5 - \alpha_5 p_{N^\sigma(5,6]} \\ &= (\alpha_{10} + \alpha_7 + \alpha_6)p_5 - \alpha_5(p_{\bar{S}_2^{\sigma_0}} + p_{\bar{S}_3^{\sigma_0}} + p_{10} + p_7 + p_6) \\ &= 14 \cdot 9 - 3 \cdot 32 = 30.\end{aligned}$$

Since $\delta_5^\sigma(6, S) > 0$ we have $l^* = 4$ and thus the processing order σ is updated as illustrated in Figure 4.5. Moreover, $v(S)$ is increased by $\delta_5^\sigma(6, S) = 30$, so $v(S) := 254$.

$\bar{S}_0^{\sigma_0}$	3	2	1	$\bar{S}_1^{\sigma_0}$	4	$\bar{S}_2^{\sigma_0}$	$\bar{S}_3^{\sigma_0}$	10	7	6	5	9	8	$\bar{S}_4^{\sigma_0}$
------------------------	---	---	---	------------------------	---	------------------------	------------------------	----	---	---	---	---	---	------------------------

Figure 4.5: The processing order σ after player 5 is considered in Example 4.1

Since the previous predecessor of player 5, player 4, is a member of coalition S , he becomes the new current player (so $i := 4$ and $k := 2$).

$k = 2, i = 4$: Since component $S_3^{\sigma_0, \sigma}$ is empty and all players in component $S_4^{\sigma_0, \sigma}$ have urgencies smaller than u_4 , we have $A := \emptyset$. This means that it is not possible to reduce the total costs by moving player 4 to a different component. Hence, σ and $v(S)$ are not changed.

Since the predecessor of player 4 is outside S , we next consider the first component ($k := 1$) and the last player of the first component ($i := 1$).

$k = 1, i = 1$: Here $A := \{2, 4\}$. According to the urgencies of the players, if player 1 is moved to a different component then the position of player 1 should be either directly behind player 4 ($t_2 := 4$) or directly behind player 10 ($t_4 := 10$). The resulting cost savings are

$$\begin{aligned}\delta_1^\sigma(4, S) &= \alpha_{S^\sigma(1,4]}p_1 - \alpha_1p_{N^\sigma(1,4]} \\ &= (\alpha_4)p_1 - \alpha_1(p_{\bar{S}_1^{\sigma_0}} + p_4) \\ &= 6 \cdot 9 - 4 \cdot 9 = 18.\end{aligned}$$

and

$$\begin{aligned}\delta_1^\sigma(10, S) &= \alpha_{S^\sigma(1,10]}p_1 - \alpha_1p_{N^\sigma(1,10]} \\ &= (\alpha_4 + \alpha_{10})p_1 - \alpha_1(p_{\bar{S}_1^{\sigma_0}} + p_4 + p_{\bar{S}_2^{\sigma_0}} + p_{\bar{S}_3^{\sigma_0}} + p_{10}) \\ &= 13 \cdot 9 - 4 \cdot 24 = 21.\end{aligned}$$

Since moving player 1 to component $S_4^{\sigma_0, \sigma}$ results in larger cost savings than moving player 1 to component $S_2^{\sigma_0, \sigma}$, we have $l^* := 4$. Therefore, processing order σ is updated as illustrated in Figure 4.6 and $v(S)$ is increased by $\delta_1^\sigma(10, S) = 21$, so $v(S) := 275$.

$\bar{S}_0^{\sigma_0}$	3	2	$\bar{S}_1^{\sigma_0}$	4	$\bar{S}_2^{\sigma_0}$	$\bar{S}_3^{\sigma_0}$	10	1	7	6	5	9	8	$\bar{S}_4^{\sigma_0}$
------------------------	---	---	------------------------	---	------------------------	------------------------	----	---	---	---	---	---	---	------------------------

Figure 4.6: The processing order σ after player 1 is considered in Example 4.1

Since the previous predecessor of player 1, player 2, is also a member of coalition S , he becomes the new current player ($i := 2$ and $k := 1$).

$k = 1, i = 2$: Like in the previous step it can be concluded that $A := \{2, 4\}$, $t_2 := 4$ and $t_4 := 10$, so the potential cost savings are

$$\begin{aligned}\delta_2^\sigma(4, S) &= \alpha_{S^\sigma(2,4]}p_2 - \alpha_2p_{N^\sigma(2,4]} \\ &= (\alpha_4)p_2 - \alpha_2(p_{\bar{S}_1^{\sigma_0}} + p_4) \\ &= 6 \cdot 6 - 3 \cdot 9 = 9.\end{aligned}$$

and

$$\begin{aligned}\delta_2^\sigma(10, S) &= \alpha_{S^\sigma(2,10]}p_2 - \alpha_2p_{N^\sigma(2,10]} \\ &= (\alpha_4 + \alpha_{10})p_2 - \alpha_2(p_{\bar{S}_1^{\sigma_0}} + p_4 + p_{\bar{S}_2^{\sigma_0}} + p_{\bar{S}_3^{\sigma_0}} + p_{10}) \\ &= 13 \cdot 6 - 3 \cdot 24 = 5.\end{aligned}$$

Since $\delta_2^\sigma(4, S) > \delta_2^\sigma(10, S)$ we have $l^* := 2$ and thus the processing order σ is modified by moving player 2 to the position directly behind player 4 (cf. Figure 4.7). Moreover, $v(S)$ is increased by $\delta_2^\sigma(4, S) = 9$, so $v(S) := 284$.

$\bar{S}_0^{\sigma_0}$	3	$\bar{S}_1^{\sigma_0}$	4	2	$\bar{S}_2^{\sigma_0}$	$\bar{S}_3^{\sigma_0}$	10	1	7	6	5	9	8	$\bar{S}_4^{\sigma_0}$
------------------------	---	------------------------	---	---	------------------------	------------------------	----	---	---	---	---	---	---	------------------------

Figure 4.7: The processing order σ after player 2 is considered in Example 4.1

The next player to be considered is player 3 ($i := 3$ and $k := 1$).

$k = 1, i = 3$: Since all players in the components $S_2^{\sigma_0, \sigma}$ and $S_4^{\sigma_0, \sigma}$ have urgency smaller than u_3 , we have $A := \emptyset$. Hence, σ and $v(S)$ are not changed.

Next $k := 0$. According to Step 4, the algorithm terminates and an optimal order for coalition S is found, namely the processing order in Figure 4.7 which can be summarized as

$$\sigma_S^{\text{alg}} = [2, 4][1, 10][5, 6][6, 7]\sigma_0^S.$$

Moreover, the total cost savings obtained are 284. ²

△

The following lemma shows that Algorithm 1 always constructs an urgency respecting admissible order for a certain coalition S . This lemma is used to prove that the processing order found by the algorithm is also optimal with respect to coalition S (see Theorem 4.3).

Lemma 4.2. *Let (N, σ_0, p, α) be a one-machine sequencing situation and let $S \in 2^N \setminus \{\emptyset\}$. Then Algorithm 1 constructs an urgency respecting admissible order σ_S^{alg} for coalition S .*

Proof. For simplicity we denote in this proof σ_S^{alg} by σ^{alg} . We only have to prove that σ^{alg} satisfies the partial tiebreaking condition, because the other conditions of admissibility and urgency respecting are satisfied trivially by construction. Suppose, for sake of contradiction, that σ^{alg} does not satisfy the partial tiebreaking condition. Then there exist $i, j \in N$, $k \in \{1, \dots, h(\sigma_0, S)\}$ with $u_i = u_j$, $i < j$, $i, j \in S_k^{\sigma_0}$ while $\sigma^{\text{alg}}(i) > \sigma^{\text{alg}}(j)$. Without loss of generality we can assume $\sigma_0^S(j) = \sigma_0^S(i) + 1$. Let σ^j be the order obtained during the run of the algorithm just before player j is considered and σ^i be the order obtained during the run of the algorithm just before player i is considered, i.e., immediately after player j is considered. We distinguish between two cases: $\sigma^j = \sigma^i$ and $\sigma^j \neq \sigma^i$.

Case 1: $\sigma^j = \sigma^i$, i.e., player j has not been moved by the algorithm. Then, due to the structure of the algorithm, there is a modified component $S_m^{\sigma_0, \sigma^i} \neq \emptyset$ and a player $t_m \in S_m^{\sigma_0, \sigma^i}$ with $u_{t_m} > u_i$, $m > k$ such that

$$\delta_i^{\sigma^i}(t_m, S) > 0,$$

while

$$\delta_j^{\sigma^j}(t_m, S) \leq 0.$$

However,

$$\begin{aligned} \delta_i^{\sigma^i}(t_m, S) &= \alpha_{S^{\sigma^i}(i, t_m]} p_i - \alpha_i p_{N^{\sigma^i}(i, t_m]} \\ &= \alpha_{S^{\sigma^j}(j, t_m]} p_i + \alpha_j p_i - \alpha_i p_{N^{\sigma^j}(j, t_m]} - \alpha_i p_j \\ &= \alpha_{S^{\sigma^j}(j, t_m]} p_i - \alpha_i p_{N^{\sigma^j}(j, t_m]} > 0, \end{aligned}$$

and thus

$$\frac{\alpha_j}{p_j} = \frac{\alpha_i}{p_i} < \frac{\alpha_{S^{\sigma^j}(j, t_m]}}{p_{N^{\sigma^j}(j, t_m]}}.$$

Therefore, $\delta_j^{\sigma^j}(t_m, S) = \alpha_{S^{\sigma^j}(j, t_m]} p_i - \alpha_i p_{N^{\sigma^j}(j, t_m]} > 0$, which is a contradiction.

Case 2: $\sigma^j \neq \sigma^i$, i.e., player j has been moved by the algorithm. Let $l > k$ be such that

²Note that the only feature of the algorithm that has not been illustrated in this example, is the tiebreaking rule in Step 2 for choosing l^* .

$j \in S_l^{\sigma_0, \sigma^i}$ and let $t_l \in S_l^{\sigma_0, \sigma^j}$ such that $\sigma^i = [j, t_l]\sigma^j$. Due to the structure of the algorithm there is an $m > l$ and $t_m \in S_m^{\sigma_0, \sigma^i}$ with

$$\delta_i^{\sigma^i}(t_m, S) > \delta_i^{\sigma^i}(t_l, S),$$

while

$$\delta_j^{\sigma^j}(t_m, S) \leq \delta_j^{\sigma^j}(t_l, S).$$

However,

$$\begin{aligned} \delta_i^{\sigma^i}(t_m, S) - \delta_i^{\sigma^i}(t_l, S) &= \alpha_{S^{\sigma^i}(i, t_m]} p_i - \alpha_i p_{N^{\sigma^i}(i, t_m]} - \alpha_{S^{\sigma^i}(i, t_l]} p_i + \alpha_i p_{N^{\sigma^i}(i, t_l]} \\ &= \alpha_{S^{\sigma^i}(t_l, t_m]} p_i - \alpha_i p_{N^{\sigma^i}(t_l, t_m]} \\ &= \alpha_{S^{\sigma^j}(t_l, t_m]} p_i + \alpha_j p_i - \alpha_i p_{N^{\sigma^j}(t_l, t_m]} - \alpha_i p_j \\ &= \alpha_{S^{\sigma^j}(t_l, t_m]} p_i - \alpha_i p_{N^{\sigma^j}(t_l, t_m]} > 0, \end{aligned}$$

and thus

$$\frac{\alpha_j}{p_j} = \frac{\alpha_i}{p_i} < \frac{\alpha_{S^{\sigma^j}(t_l, t_m]}}{p_{N^{\sigma^j}(t_l, t_m]}}.$$

Therefore,

$$\begin{aligned} \delta_j^{\sigma^j}(t_m, S) - \delta_j^{\sigma^j}(t_l, S) &= \alpha_{S^{\sigma^j}(j, t_m]} p_j - \alpha_j p_{N^{\sigma^j}(j, t_m]} - \alpha_{S^{\sigma^j}(j, t_l]} p_j + \alpha_j p_{N^{\sigma^j}(j, t_l]} \\ &= \alpha_{S^{\sigma^j}(t_l, t_m]} p_j - \alpha_j p_{N^{\sigma^j}(t_l, t_m]} > 0, \end{aligned}$$

which is a contradiction. $\square$

The class of urgency respecting orders for S can be divided into groups with identical costs for S . Within these groups one can select a unique representative σ by imposing that

(i) for all $i, j \in S_k^{\sigma_0, \sigma}$ with $k \in \{1, \dots, h(\sigma_0, S)\}$:

$$u_i = u_j, \sigma_0^S(i) < \sigma_0^S(j) \Rightarrow \sigma(i) < \sigma(j),$$

(ii) there does not exist a different urgency respecting order $\hat{\sigma}$ with $\sum_{i \in S} \alpha_i C_i(\hat{\sigma}) = \sum_{i \in S} \alpha_i C_i(\sigma)$ and $i \in S$ such that

$$i \in S_k^{\sigma_0, \hat{\sigma}} \text{ and } i \in S_l^{\sigma_0, \sigma} \text{ for some } k, l \in \{1, \dots, h(\sigma_0, S)\} \text{ with } k < l,$$

and

$$j \in S_k^{\sigma_0, \hat{\sigma}} \text{ and } j \in S_k^{\sigma_0, \sigma} \text{ for some } k \in \{1, \dots, h(\sigma_0, S)\}$$

for all $j \in N \setminus \{i\}$.

Thus, condition (i) determines the relative order for players with the same urgency and from the same modified component of S with respect to σ . Likewise, condition (ii) ensures that no player in S can be moved to an earlier modified component of σ while the total costs remain the same. In particular, for coalition S there is a unique representative of the urgency respecting optimal orders and denote this order by σ_S^* .

Theorem 4.3. *Let (N, σ_0, p, α) be a one-machine sequencing situation and let $S \in 2^N \setminus \{\emptyset\}$. Then $\sigma_S^{alg} = \sigma_S^*$.*

Proof. Suppose for sake of contradiction $\sigma_S^{\text{alg}} \neq \sigma_S^*$. For simplicity we denote in this proof σ_S^{alg} by σ^{alg} and σ_S^* by σ^* . From Lemma 4.2 it follows that σ^{alg} is an urgency respecting admissible order for coalition S and thus there is a well-defined procedure of consecutive movements to go from initial order σ_0 to the urgency respecting order σ^{alg} , as is also the case for order σ^* . Therefore, we can distinguish between the following three cases:

1. $|R(\sigma^{\text{alg}})| \geq 1$ and $|R(\sigma^*)| = 0$, i.e., at least one player switched component in σ^{alg} , but no player switched component in σ^* ,
2. $|R(\sigma^{\text{alg}})| = 0$ and $|R(\sigma^*)| \geq 1$, i.e., no player switched component in σ^{alg} , but at least one player switched component in σ^* ,
3. $|R(\sigma^{\text{alg}})| \geq 1$ and $|R(\sigma^*)| \geq 1$, i.e., both in σ^{alg} and σ^* there is at least one player who switched component.

Note that the case where no player switched component in both σ^{alg} and σ^* , i.e., $|R(\sigma^{\text{alg}})| = |R(\sigma^*)| = 0$, is not possible because then it would hold that $\sigma^{\text{alg}} = \sigma^* (= \sigma_0^S)$.

Case 1: Denote $\tau = \sigma^* (= \tau^{\sigma^{\text{alg}}, S, 0})$, $r = r_1(\sigma^{\text{alg}})$ and $m = m_1(\sigma^{\text{alg}})$. The algorithm moved player r to the position behind player m , therefore $\delta_r^\tau(m, S) > 0$. Consequently, $\delta_r^{\sigma^*}(m, S) > 0$ which is a contradiction with the optimality of σ^* .

Case 2: Denote $\tau = \sigma^{\text{alg}} (= \tau^{\sigma^*, S, 0})$, $\tau^* = \tau^{\sigma^*, S, 1}$, $r = r_1(\sigma^*)$ and $m = m_1(\sigma^*)$. The algorithm did not move player r to the position behind player m , therefore $\delta_r^\tau(m, S) \leq 0$, i.e.,

$$\frac{\alpha_r}{p_r} \geq \frac{\alpha_{S^\tau(r, m)}}{p_{N^\tau(r, m)}} = \frac{\alpha_{S^{\tau^*}[k, r]}}{p_{N^{\tau^*}[k, r]}},$$

where player $k \in N$ is the direct follower of player r with respect to τ . Note that $\bar{S}^{\tau^*}[k, r] = \bar{S}^{\sigma^*}[k, r]$ and $S^{\tau^*}[k, r] \subset S^{\sigma^*}[k, r]$ because every follower of player k with respect to τ^* will not be moved anymore. If $S^{\tau^*}[k, r] = S^{\sigma^*}[k, r]$, then

$$\frac{\alpha_r}{p_r} \geq \frac{\alpha_{S^{\sigma^*}[k, r]}}{p_{N^{\sigma^*}[k, r]}},$$

i.e., $\delta_r^{\sigma^*}(k, S) \geq 0$, which is either a contradiction with the optimality of σ^* or with condition (ii) of σ^* . Hence, $S^{\tau^*}[k, r] \subsetneq S^{\sigma^*}[k, r]$. Note that $S^{\sigma^*}[k, r] \setminus S^{\tau^*}[k, r]$ consists of players positioned between players k and r with respect to σ^* , but who are in front of player r with respect to τ^* . Let us call these players the “new players”. Define $q \in S^{\sigma^*}[k, r] \setminus S^{\tau^*}[k, r]$ such that

$$\sigma^*(q) \leq \sigma^*(i),$$

for all $i \in S^{\sigma^*}[k, r] \setminus S^{\tau^*}[k, r]$. This means that player q is among the “new players” the player positioned first with respect to σ^* and thus $S^{\sigma^*}[k, q]$ does not contain any “new players”. We show that moving player q in front of player k with respect to σ^* does not result in worse costs for coalition S which is a contradiction. As $\tau(q) < \tau(r) < \tau(k)$ and $\sigma^*(k) < \sigma^*(q) < \sigma^*(r)$, we know that the swap of players q and r with respect to σ^* is admissible. Consequently, from Lemma 4.1 it follows that $u_q \geq u_r$. If $N^{\tau^*}[k, r] = N^{\sigma^*}[k, q]$, then

$$\frac{\alpha_q}{p_q} \geq \frac{\alpha_r}{p_r} \geq \frac{\alpha_{S^{\tau^*}[k, r]}}{p_{N^{\tau^*}[k, r]}} = \frac{\alpha_{S^{\sigma^*}[k, q]}}{p_{N^{\sigma^*}[k, q]}},$$

i.e., $\delta_q^{\sigma^*}(k, S) \geq 0$, which is either a contradiction with the optimality of σ^* or with condition (ii) of σ^* . Hence, $N^{\tau^*}[k, r] \neq N^{\sigma^*}[k, q]$, i.e., $N^{\tau^*}[k, r] \cap F(\sigma^*, q) \neq \emptyset$. Define $t \in N^{\tau^*}[k, r] \cap F(\sigma^*, q)$ such that

$$\sigma^*(t) \leq \sigma^*(i),$$

for all $i \in N^{\tau^*}[k, r] \cap F(\sigma^*, q)$. Hence, player $t \in N^{\tau^*}[k, r]$ is defined such that $N^{\tau^*}[k, t] = N^{\sigma^*}[k, q]$. This property of player t is used in order to show the contradiction. Denote by $\hat{\tau}$ the processing order that is obtained from τ^* by moving player r to the position directly in front of player t , which is the same as $[r, l]\tau$ where player $l \in N$ is the direct predecessor of player t with respect to τ . Note that since $q \in R(\sigma^*)$ and $\sigma^* \in \mathcal{A}(\sigma_0, S)$, we know from condition (ii) of admissibility that $l \in S$ and thus $\hat{\tau} \in \mathcal{A}(\sigma_0, S)$. Moreover, as the algorithm did not move player r we know $\delta_r^{\tau}(l, S) \leq 0$, and thus

$$\frac{\alpha_q}{p_q} \geq \frac{\alpha_r}{p_r} \geq \frac{\alpha_{S^{\tau}(r, l)}}{p_{N^{\tau}(r, l)}} = \frac{\alpha_{S^{\hat{\tau}}[k, r]}}{p_{N^{\hat{\tau}}[k, r]}} = \frac{\alpha_{S^{\sigma^*}[k, q]}}{p_{N^{\sigma^*}[k, q]}},$$

where the last equality follows from $N^{\hat{\tau}}[k, r] = N^{\tau^*}[k, t] = N^{\sigma^*}[k, q]$. Consequently, $\delta_q^{\sigma^*}(k, S) \geq 0$ which is a contradiction with the optimality of σ^* , or with condition (i) or (ii) of σ^* . Note that if player t does not exist, i.e., $N^{\tau^*}[k, r] \cap F(\sigma^*, q) = \emptyset$, then

Case 3: We make a further distinction, namely the following three cases:

3.1. $|R(\sigma^{\text{alg}})| > |R(\sigma^*)|$ and

$$[r_k(\sigma^{\text{alg}}), m_k(\sigma^{\text{alg}})] = [r_k(\sigma^*), m_k(\sigma^*)],$$

for all $k \in \{1, \dots, |R(\sigma^*)|\}$. Hence, there are more players in σ^{alg} switching components than in σ^* . Moreover, σ^* is an intermediate processing order of σ^{alg} in the procedure of consecutive movements to go from σ_0 to σ^{alg} .

3.2. $|R(\sigma^{\text{alg}})| < |R(\sigma^*)|$ and

$$[r_k(\sigma^{\text{alg}}), m_k(\sigma^{\text{alg}})] = [r_k(\sigma^*), m_k(\sigma^*)],$$

for all $k \in \{1, \dots, |R(\sigma^{\text{alg}})|\}$. Hence, there are more players in σ^* switching components than in σ^{alg} . Moreover, σ^{alg} is an intermediate processing order of σ^* in the procedure of consecutive movements to go from σ_0 to σ^* .

3.3. There exists a $c \leq \min\{|R(\sigma^{\text{alg}})|, |R(\sigma^*)|\}$ such that

$$[r_c(\sigma^{\text{alg}}), m_c(\sigma^{\text{alg}})] \neq [r_c(\sigma^*), m_c(\sigma^*)]$$

and

$$[r_k(\sigma^{\text{alg}}), m_k(\sigma^{\text{alg}})] = [r_k(\sigma^*), m_k(\sigma^*)],$$

for all $k < c$. Hence, neither σ^{alg} is an intermediate processing order of σ^* nor σ^* is an intermediate processing order of σ^{alg} in the procedure of consecutive movements to go from σ_0 to σ^* and σ^{alg} .

Note that the case where $|R(\sigma^{\text{alg}})| = |R(\sigma^*)|$ and

$$[r_k(\sigma^{\text{alg}}), m_k(\sigma^{\text{alg}})] = [r_k(\sigma^*), m_k(\sigma^*)],$$

for all $k \in \{1, \dots, |R(\sigma^{\text{alg}})|\}$ is not possible because then it would hold that $\sigma^{\text{alg}} = \sigma^*$.

Case 3.1: Denote $\tau = \sigma^*(= \tau^{\sigma^{\text{alg}}, S, |R(\sigma^*)|})$, $r = r_{|R(\sigma^*)|+1}(\sigma^{\text{alg}})$ and $m = m_{|R(\sigma^*)|+1}(\sigma^{\text{alg}})$. Similar to Case 1 we have $\delta_r^{\tau^*}(m, S) = \delta_r^{\tau}(m, S) > 0$ which is a contradiction with the optimality of σ^* .

Case 3.2: Denote $\tau = \sigma^{\text{alg}}(= \tau^{\sigma^*, S, |R(\sigma^{\text{alg}})|})$, $\tau^* = \tau^{\sigma^*, S, |R(\sigma^{\text{alg}})|+1}$, $r = r_{|R(\sigma^{\text{alg}})|+1}(\sigma^*)$ and $m = m_{|R(\sigma^{\text{alg}})|+1}(\sigma^*)$. Then, using the same arguments as in Case 2, we have a contradiction.

Case 3.3: We make a further distinction, namely the following three cases:

- 3.3.1. $\sigma_0^S(r_c(\sigma^{\text{alg}})) > \sigma_0^S(r_c(\sigma^*))$, i.e., player $r_c(\sigma^{\text{alg}})$ switched component in σ^{alg} but did not switch component in σ^* .
- 3.3.2. $\sigma_0^S(r_c(\sigma^{\text{alg}})) < \sigma_0^S(r_c(\sigma^*))$, i.e., player $r_c(\sigma^*)$ switched component in σ^* but did not switch component in σ^{alg} .
- 3.3.3. $r_c(\sigma^{\text{alg}}) = r_c(\sigma^*)$, $m_c(\sigma^{\text{alg}}) \neq m_c(\sigma^*)$, i.e., player $r_c(\sigma^{\text{alg}})(= r_c(\sigma^*))$ switched component in both σ^{alg} and σ^* , but he is positioned behind two different players.

Case 3.3.1: Denote $\tau^* = \tau^{\sigma^*, S, c-1}(= \tau^{\sigma^{\text{alg}}, S, c-1})$, $r = r_c(\sigma^{\text{alg}})$ and $m = m_c(\sigma^{\text{alg}})$. The algorithm moved player r to the position behind player m , therefore $\delta_r^{\tau^*}(m, S) > 0$, i.e.,

$$\frac{\alpha_r}{p_r} < \frac{\alpha_{S^{\tau^*}(r, m]}}{p_{N^{\tau^*}(r, m]}}.$$

Note that $\bar{S}^{\tau^*}(r, m] = \bar{S}^{\sigma^*}(r, m]$ and $S^{\tau^*}(r, m] \subset S^{\sigma^*}(r, m]$ because every follower of player r with respect to τ^* will not be moved anymore. If $S^{\tau^*}(r, m] = S^{\sigma^*}(r, m]$, then

$$\frac{\alpha_r}{p_r} < \frac{\alpha_{S^{\sigma^*}(r, m]}}{p_{N^{\sigma^*}(r, m]}},$$

i.e., $\delta_r^{\sigma^*}(m, S) > 0$, which is a contradiction with the optimality of σ^* . Hence, $S^{\tau^*}(r, m] \subsetneq S^{\sigma^*}(r, m]$. Note that $S^{\sigma^*}(r, m] \setminus S^{\tau^*}(r, m]$ consists of players positioned between players r and m with respect to σ^* , but who are in front of player r with respect to τ^* . Let us call these players the “new players”. Define $q \in S^{\sigma^*}(r, m] \setminus S^{\tau^*}(r, m]$ such that

$$\sigma^*(q) \geq \sigma^*(i),$$

for all $i \in S^{\sigma^*}(r, m] \setminus S^{\tau^*}(r, m]$. This means that player q is among the “new players” the player positioned last with respect to σ^* and thus $S^{\sigma^*}(q, m]$ does not contain any “new players”. We show that moving player q behind player m with respect to σ^* results in cost savings for coalition S which is a contradiction. As $\sigma_0(q) < \sigma_0(r)$ and $\sigma^*(q) > \sigma^*(r)$, we know that the swap of players q and r with respect to σ^* is admissible. Consequently, from Lemma 4.1 it follows that $u_q \leq u_r$. If $N^{\tau^*}(r, m] = N^{\sigma^*}(q, m]$, then

$$\frac{\alpha_q}{p_q} \leq \frac{\alpha_r}{p_r} < \frac{\alpha_{S^{\tau^*}(r, m]}}{p_{N^{\tau^*}(r, m]}} = \frac{\alpha_{S^{\sigma^*}(q, m]}}{p_{N^{\sigma^*}(q, m]}},$$

i.e., $\delta_q^{\sigma^*}(m, S) > 0$, which is a contradiction with the optimality of σ^* . Hence, $N^{\tau^*}(r, m] \neq N^{\sigma^*}(q, m]$, i.e., $N^{\tau^*}(r, m] \cap P(\sigma^*, q) \neq \emptyset$. Define $t \in N^{\tau^*}(r, m] \cap P(\sigma^*, q)$ such that

$$\sigma^*(t) \geq \sigma^*(i),$$

for all $i \in N^{\tau^*}(r, m] \cap P(\sigma^*, q)$. Hence, player $t \in N^{\tau^*}(r, m]$ is defined such that $N^{\tau^*}(t, m] = N^{\sigma^*}(q, m]$. This property of player t is used in order to show the contradiction. Observe that

$$\delta_r^{\tau^*}(m, S) = \delta_r^{\tau^*}(t, S) + \delta_r^{\hat{\tau}}(m, S),$$

where $\hat{\tau}$ is the processing order that is obtained from τ^* by moving player r to the position directly behind player t . Note that since $q \in R(\sigma^*)$ and $\sigma^* \in \mathcal{A}(\sigma_0, S)$, we know from condition (ii) of admissibility that $t \in S$ and thus $\hat{\tau} \in \mathcal{A}(\sigma_0, S)$. Since the algorithm moved player r behind player m and because $\tau(t) < \tau(m)$, we know from the greedy aspect of the algorithm that $\delta_r^{\tau^*}(m, S) > \delta_r^{\tau^*}(t, S)$, i.e., $\delta_r^{\hat{\tau}}(m, S) > 0$, and thus

$$\frac{\alpha_q}{p_q} \leq \frac{\alpha_r}{p_r} < \frac{\alpha_{S^{\hat{\tau}}(r, m]}}{p_{N^{\hat{\tau}}(r, m]}}} = \frac{\alpha_{S^{\sigma^*}(q, m]}}{p_{N^{\sigma^*}(q, m]}}},$$

where the last equality follows from $N^{\hat{\tau}}(r, m] = N^{\tau^*}(t, m] = N^{\sigma^*}(q, m]$. Consequently, $\delta_q^{\sigma^*}(m, S) > 0$ which is a contradiction with the optimality of σ^* . Note that if player t does not exist, i.e., $N^{\tau^*}(r, m] \cap P(\sigma^*, q) = \emptyset$, then $N^{\tau^*}(r, m] = N^{\sigma^*}(q, m]$, so

$$\frac{\alpha_q}{p_q} \leq \frac{\alpha_r}{p_r} < \frac{\alpha_{S^{\tau^*}(r, m]}}{p_{N^{\tau^*}(r, m]}}} = \frac{\alpha_{S^{\sigma^*}(q, m]}}{p_{N^{\sigma^*}(q, m]}}},$$

i.e., $\delta_q^{\sigma^*}(m, S) > 0$, which is again a contradiction with the optimality of σ^* .

Case 3.3.2: Denote $\tau = \tau^{\sigma^{\text{alg}}, S, c-1} (= \tau^{\sigma^*, S, c-1})$, $\tau^* = \tau^{\sigma^*, S, c}$, $r = r_c(\sigma^*)$ and $m = m_c(\sigma^*)$. Then, using the same arguments as in Case 2, we have a contradiction.

Case 3.3.3: We make a further distinction, namely the following three cases:

- 3.3.3.1. $m_c(\sigma^{\text{alg}}) \in S_k^{\sigma_0, \sigma^{\text{alg}}}$ and $m_c(\sigma^*) \in S_l^{\sigma_0, \sigma^*}$ with $k > l$, i.e., player $r_c(\sigma^{\text{alg}}) (= r_c(\sigma^*))$ is moved with respect to σ^{alg} to a modified component further than with respect to σ^* .
- 3.3.3.2. $m_c(\sigma^{\text{alg}}) \in S_k^{\sigma_0, \sigma^{\text{alg}}}$ and $m_c(\sigma^*) \in S_l^{\sigma_0, \sigma^*}$ with $k < l$, i.e., player $r_c(\sigma^{\text{alg}}) (= r_c(\sigma^*))$ is moved with respect to σ^* to a modified component further than with respect to σ^{alg} .
- 3.3.3.3. $m_c(\sigma^{\text{alg}}) \in S_k^{\sigma_0, \sigma^{\text{alg}}}$ and $m_c(\sigma^*) \in S_k^{\sigma_0, \sigma^*}$, i.e., player $r_c(\sigma^{\text{alg}}) (= r_c(\sigma^*))$ is moved both with respect to σ^{alg} and σ^* to the same modified component.

Case 3.3.3.1: Denote $\bar{\tau} = \tau^{\sigma^*, S, c-1} (= \tau^{\sigma^{\text{alg}}, S, c-1})$, $\tau^* = \tau^{\sigma^*, S, c}$, $r = r_c(\sigma^{\text{alg}}) (= r_c(\sigma^*))$ and $m = m_c(\sigma^{\text{alg}})$. Observe that

$$\delta_r^{\bar{\tau}}(m, S) = \delta_r^{\bar{\tau}}(m_c(\sigma^*), S) + \delta_r^{\tau^*}(m, S),$$

due to the fact that $\tau^* = [r, m_c(\sigma^*)]\bar{\tau}$. Since the algorithm moved player r behind player m and because $\bar{\tau}(m_c(\sigma^*)) < \bar{\tau}(m)$, we know from the greedy aspect of the algorithm that $\delta_r^{\bar{\tau}}(m, S) > \delta_r^{\bar{\tau}}(m_c(\sigma^*), S)$, i.e., $\delta_r^{\tau^*}(m, S) > 0$. Then, using the same arguments as in Case 3.3.1, we have a contradiction.

Case 3.3.3.2: Denote $\bar{\tau} = \tau^{\sigma^{\text{alg}}, S, c-1} (= \tau^{\sigma^*, S, c-1})$, $\tau^* = \tau^{\sigma^*, S, c}$, $\tau = \tau^{\sigma^{\text{alg}}, S, c}$, $r = r_c(\sigma^*) (= r_c(\sigma^{\text{alg}}))$ and $m = m_c(\sigma^*)$. Observe that

$$\delta_r^{\bar{\tau}}(m, S) = \delta_r^{\bar{\tau}}(m_c(\sigma^{\text{alg}}), S) + \delta_r^{\tau}(m, S),$$

because $\tau = [r, m_c(\sigma^{\text{alg}})]\bar{\tau}$. Since the algorithm moved player r behind player $m_c(\sigma^{\text{alg}})$, we know from the greedy aspect of the algorithm that $\delta_r^{\bar{\tau}}(m_c(\sigma^{\text{alg}}), S) \geq \delta_r^{\bar{\tau}}(m, S)$, i.e., $\delta_r^{\tau}(m, S) \leq 0$. Then, using the same arguments as in Case 2, we have a contradiction.

Case 3.3.3.3: Denote $r = r_c(\sigma^{\text{alg}})(= r_c(\sigma^*))$. Since $\tau^{\sigma^{\text{alg}}, S, c}$ and $\tau^{\sigma^*, S, c}$ are both urgency respecting, we know $u_r \leq u_{m_c(\sigma^{\text{alg}})}$ and $u_r \leq u_{m_c(\sigma^*)}$. Moreover, by the definition of the algorithm we know $u_r < u_{m_c(\sigma^{\text{alg}})}$. Additionally, since $m_c(\sigma^{\text{alg}}) \neq m_c(\sigma^*)$, it must hold that $u_r = u_{m_c(\sigma^*)}$. Consequently, since $\sigma_0^S(r) < \sigma_0^S(m_c(\sigma^*))$ and $\sigma^*(r) > \sigma_0^S(m_c(\sigma^*))$, we have a contradiction with condition (i) of σ^* .

As one can see, for every (sub) case there is a contradiction and thus $\sigma^{\text{alg}} = \sigma^*$. $\square$

5 Step out sequencing games

Another example of relaxed sequencing game is the so-called *Step out sequencing game*. In a Step out sequencing game a member of a coalition S can swap with another member of the coalition S who is in the same component of S with respect to σ_0 , while he is also allowed to step out from his initial position in the processing order and enter at the rear of the processing order. Hence, in contrast to a SoSi in sequencing game, a player in S cannot join any position later in the processing order but only the rear of the processing order. Example 5.1 illustrates the differences between admissible orders in a Step out sequencing game and a SoSi sequencing game.

Example 5.1. Consider a one-machine sequencing situation (N, σ_0, p, α) with $S = \{1, 2, 3\}$. Assume that the components of coalition S with respect to σ_0 are given by

$$S_1^{\sigma_0} = \{1\} \text{ and } S_2^{\sigma_0} = \{2, 3\}.$$

Moreover, assume that $\bar{S}_2^{\sigma_0} \neq \emptyset$ (cf. Figure 5.1(a)). In a SoSi sequencing game it is allowed for player 1 to step out from his position in the processing order and to step in in the second component of S , for example in between players 2 and 3. However, this is not allowed in a Step out sequencing game, because in a Step out sequencing game player 1 can only enter at the rear of the processing order. This means that processing order $\hat{\sigma}$ in Figure 5.1(b) is admissible in a SoSi sequencing game but not in a Step out sequencing game. $\triangle$

Figure 5.1: The processing orders used in Example 5.1

The upper bound on the value of a coalition in Lemma 3.1 is also valid for Step out sequencing games. This can be shown by using similar arguments as in the proof of Lemma 3.1. Therefore, it follows from the arguments in the proof of Theorem 3.2 that for every one-machine sequencing situation (N, σ_0, p, α) the corresponding Step out sequencing game has a non-empty core.

Finding a polynomial time algorithm determining an optimal order for every possible coalition for a Step out sequencing game turns out to be different than for a SoSi sequencing game. Example 5.2 gives an example of a Step out sequencing game where moving some players individually is not beneficial, but moving these players simultaneously is. This illustrates that the algorithm of SoSi sequencing games cannot be applied to Step out sequencing games because this algorithm moves players individually.

Example 5.2. Reconsider the one-machine sequencing situation (N, σ_0, p, α) of Example 5.1 with $S = \{1, 2, 3\}$. In Figure 5.2 an illustration can be found of an initial order σ_0 together with the cost coefficients and processing times (the numbers above and below the players, respectively). The completion times of the players with respect to this initial order are also indicated in the figure (bottom line in bold).

Some alternative admissible orders are given in Figure 5.2. The corresponding cost savings for coalition S with respect to these admissible orders are as follows

$$\begin{aligned} \sum_{i \in \{1,2,3\}} \alpha_i (C_i(\sigma_0) - C_i(\sigma_1)) &= 0 + 0 - 1 = -1, \\ \sum_{i \in \{1,2,3\}} \alpha_i (C_i(\sigma_0) - C_i(\sigma_2)) &= -105 + 72 + 8 = -25, \\ \sum_{i \in \{1,2,3\}} \alpha_i (C_i(\sigma_0) - C_i(\sigma_3)) &= -49 + 72 - 1 = 22. \end{aligned}$$

Hence, both moving player 1 and player 3 individually to the rear of the processing order is not beneficial (cf. σ_1 and σ_2 , respectively). However, if you move players 1 and 3 simultaneously to the rear of the processing order then some cost savings are obtained (cf. σ_3). This illustrates that a more sophisticated algorithm is needed. $\triangle$

		7			9			1			α_i
$\bar{S}_0^{\sigma_0}$	1	$\bar{S}_1^{\sigma_0}$	2	3	$\bar{S}_2^{\sigma_0}$						σ_0
1	8	1	5	8	1						p_i
1	9	10	15	23	24						$C_i(\sigma_0)$

(a)

		7			9			1			α_i
$\bar{S}_0^{\sigma_0}$	1	$\bar{S}_1^{\sigma_0}$	2	$\bar{S}_2^{\sigma_0}$	3						σ_1
1	8	1	5	1	8						p_i
1	9	10	15	16	24						$C_i(\sigma_1)$

(b)

		9			1			7			α_i
$\bar{S}_0^{\sigma_0}$	$\bar{S}_1^{\sigma_0}$	2	3	$\bar{S}_3^{\sigma_0}$	1						σ_2
1	1	5	8	1	8						p_i
1	2	7	15	16	24						$C_i(\sigma_2)$

(c)

		9			7			1			α_i
$\bar{S}_0^{\sigma_0}$	$\bar{S}_1^{\sigma_0}$	2	$\bar{S}_3^{\sigma_0}$	1	3						σ_3
1	1	5	1	8	8						p_i
1	2	7	8	16	24						$C_i(\sigma_3)$

(d)

Figure 5.2: Various processing orders used in Example 5.2

Acknowledgments: The authors wish to thank John Kleppe for his inspirational remarks that led to the idea of Step out sequencing games.

References

Curiel, I., H. Hamers, and F. Klijn (2002). Sequencing games: a survey. In P. Borm and H. Peters (Eds.), *Chapters in game theory, in honour of Stef Tijs*, pp. 27–50. Kluwer

Academic Publishers, The Netherlands.

- Curiel, I., G. Pederzoli, and S. Tijs (1989). Sequencing games. *European Journal of Operational Research*, **40**, 344–351.
- Curiel, I., J. Potters, R. Prasad, S. Tijs, and B. Veltman (1993). Cooperation in one machine scheduling. *Zeitschrift für Operations Research*, **38**, 113–129.
- Slikker, M. (2006). Relaxed sequencing games have a nonempty core. *Naval Research Logistics*, **53**, 235–242.
- Smith, W. (1956). Various optimizers of single-stage production. *Naval Research Logistics Quarterly*, **3**, 59–66.
- Velzen, B. van and H. Hamers (2003). On the balancedness of relaxed sequencing games. *Mathematical Methods of Operations Research*, **57**, 287–297.