Optimal threshold policies in a workload model with a variable number of service phases per job
Brouns, G.A.J.F.; van der Wal, J.

Published: 01/01/2000

Document Version
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

• A submitted manuscript is the author's version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

Citation for published version (APA):

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.

Download date: 09. Jan. 2019
Optimal threshold policies in a workload model with a variable number of service phases per job

Gido A.J.F. Brouns  
ace@win.tue.nl

Jan van der Wal  
jan.v.d.wal@tue.nl

Eindhoven University of Technology  
Department of Mathematics and Computing Science  
PO Box 513, 5600 MB Eindhoven  
The Netherlands

December 20, 2000

Abstract

We consider a basic model for two essential on-line decisions that have to be taken in workload models. The first is the decision to either continue or abort the service of a job. The second concerns the decision to either accept or reject new jobs. We show that under certain regularity conditions, there exist optimal threshold policies for these two types of decisions.

Key words: workload models, admission and termination control, optimal threshold policies, Markov decision processes.

1 Introduction

Workload models are used for performance evaluation and performance optimization of queueing systems. The basic principle of performance optimization is the question of how to control some type of system in an optimal way within the freedom provided to the decision maker. Before introducing the model (and, consequently, the type of control) we study in this paper, we review briefly the types of control that have been investigated in literature so far.

Stidham and Weber [4] present a comprehensive overview of models for the optimal control of queueing systems. Emphasis is laid on models based on Markov decision theory (i.e., dynamic programming) and on the characterization of the structure of optimal control policies. They provide an extensive list of references to literature devoted to the analysis of specific workload models.

*corresponding author: Gido A.J.F. Brouns
Topics include optimal control of service rates, optimal admission control, optimal routing control, optimal server allocation and optimal scheduling in networks of queues. Another topic—which is not mentioned in [4], and which only involves static (i.e., one-time) decision making—is the optimal order of queues in networks of queues (e.g., see Whitt [5]). A common feature of the respective models is that admission is final, i.e., once new work has been accepted for service, it must be processed by the system, and must be processed entirely, before it can be considered to be out of the system.

**Our model** The workload model we study incorporates a feature that (as far as we know) has not previously been addressed in literature, namely the issue of what we term optimal termination control. This involves making on-line decisions on whether to quit processing work-in-process and, if so, on how much work to cancel. One can imagine that the presence of some job J that has initially been accepted—e.g., because the workload was rather low at that time—can at a certain point in time turn out to be a burden to the system as a whole—e.g., because the workload has increased rapidly since the admittance of J. In such a case, one might rather dispose of J. Whether or not to cancel work will depend on the pressure of work on the one hand, and the benefit of carrying on with and finishing work-in-process on the other.

We will show that in our model, which displays both admission and termination control, both the optimal admission control policy and the optimal termination control policy are of a threshold type. This holds under some regularity conditions on the function that represents (i.e., quantifies) the benefit of executing portions of work or of executing additional work. By means of some counterexamples, it will be shown that the policies need not be of a threshold type if these regularity conditions are violated.

## 2 Model description

We study a queueing system consisting of one single server station, operating according to the FCFS discipline and possessing a buffer that is infinitely large. Jobs arrive at this station according to a Poisson process with arrival rate $\lambda \geq 0$. The workload of a job is Erlang distributed with $N \geq 1$ phases. Each phase takes an exponential time with finite positive mean $1/\mu$.

At any time one may decide to quit serving a job. The reward for a job depends on the number of service phases that have been completed. The reward corresponding to the completion of $\kappa$ phases is $r(\kappa)$, $0 \leq \kappa \leq N$, where $\kappa = 0$ means the job has not (yet) passed the first phase, possibly because it has not been taken into service at all.

We say that a job being serviced 'resides in node $k$' if it has passed through $k$ service phases and, consequently, is in its $(k+1)$th phase, $0 \leq k \leq N$. After the completion of this $(k+1)$th phase, the job moves on from node $k$ to node $k + 1$, provided service is not aborted. The maximum number of service phases is $N$.

**Remark 1**

*For technical reasons, we specify that if a job has received maximum service, and, as a consequence, has reached (final) node $N$, then it can be either maintained in the system or terminated. Maintaining means that the job remains in node $N$, occupying the server, until*
it is (finally) decided to terminate the job after all. Maintaining a job does not influence its reward, i.e., the reward that can be cashed remains $r(N)$.

Fundamental Assumption 1

[STRUCTURE OF THE REWARD FUNCTION]

The reward function $r(k)$ is non-decreasing and concave in $k$.

Non-decreasingness means that putting more work into a job does not leave us with a lower overall reward for this job. Concavity means the longer we work on a job the less rewarding it becomes to continue. All rewards are assumed to be finite.

Apart from these rewards there are holding costs for the jobs residing in the system, either waiting to be taken into service or being serviced. We assume these costs are linear in the number of jobs, viz, $ih \geq 0$ per period of time when there are $i$ jobs present. In addition, each time a job is admitted to the system, consideration costs $c \geq 0$ are incurred. If an arriving job is not admitted, then a reward $r(0)$ is earned, so $-r(0)$ can be seen as the rejection costs.

We further discount at a rate $\alpha \geq 0$, i.e., rewards and costs at time $t$ are to be multiplied by $\exp(-\alpha t)$. We treat the discount rate $\alpha$ as the rate by which the process vanishes. In other words, the process will live for an exponential time with rate $\alpha$, after which there will be no more arrivals, services, rewards or costs.

The state of the system is described by the tuple $(i, k)$, where $i$ is the number of jobs in the system and $k$ is the node the job under service resides in. If $i = 0$, then $k$ is indefinite. To indicate this, the empty system is denoted by $(0, \cdot)$.

The system evolves at arrival times, at service completion times, and eventually at the time the process vanishes. Applying the uniformization method, we can consider that transitions occur at the jump times of a Poisson process with rate $\lambda + \mu + \alpha > 0$. By scaling time, we can take $\lambda + \mu + \alpha = 1$ without loss of generality. Then, with probability $\lambda \geq 0$ a transition concerns an arrival, with probability $\mu > 0$ a service completion and with probability $\alpha \geq 0$ the ending of the process. A service completion is either a real service completion, in which case the job under service jumps from some node $k$ to node $k + 1$, or an artificial service completion, in which case the state of the system remains unaltered, viz, $(0, \cdot)$ or $(i, N)$ for some $i \geq 1$.

Because of uniformization, times between consecutive events are identically distributed. Such times are called periods and if we reverse the direction of time, we can consider the number $n$ of periods left until the process hits time zero. If the process vanishes before $n = 0$, say at $n = n_0$, then the state of the system will see no more changes during the remaining $n_0$ periods, and there will be no more rewards and costs.

Uniformization enables us to use induction on the remaining number of periods to prove our results for any finite time horizon. These results can then be extended to the infinite time horizon case; cf section 6.

The objective is to maximize the expected (discounted) profit over an $n$-period time horizon, where profit is defined as reward minus cost. We allow $\lambda > \mu/N$, as well as $\lambda = 0$. In the latter case, there is an initial number of jobs that await service, i.e., a batch, without any future arrivals.
In the next section, we indicate what the set of possible policies consists of, by giving an overview of the decisions that can be made in our model. This is followed by a mathematical formulation of the problem in terms of dynamic programming relations. The dynamic programming approach takes a prominent position in our research. However, at one particular point, we make use of the sample path approach to demonstrate a desired property. For an exposition of the sample path approach, see Liu, Nain and Towsley [3].

2.1 Decisions and decision times

The times at which decisions are to be made are the times just after an event, either concerning the arrival of a job (an arrival event) or the completion of a service phase (a service completion event). Note that this is not a restriction, because periods (times between events) are exponentially distributed. In addition, note that because of exponential event times, we only need to consider Markov policies, i.e., the decision taken at a decision time $t$ only depends on the state at time $t$ and not on the course of the process before time $t$.

**Arrival events** If the event is an arrival event, then, first of all, we have the option to either accept or reject the job seeking service. Next, we have the option to either continue or abort the service of the job under service, if there is any. If the job under service resides in node $N$, then continuing means maintaining. If there are no jobs, then service is continued per definition. If there is at least one job in the system, then termination of this job, which resides in node $k$ for some $0 \leq k \leq N$, yields a reward $r(k)$.

Jobs residing in the queue may not be aborted at this point in time. These jobs were admitted to the system earlier in time with the purpose to receive at least some attention and this may only be prevented by time hitting zero or by the process vanishing before time hits zero. Thus, if there are at least two jobs in the system, then the decision to abort service also concerns commencing service of the foremost job in the queue. Therefore, abort actually stands for abort followed by continue at the same point in time.

Note that receiving attention, i.e., being under service, does not guarantee completion of any service phases.

Jobs that are terminated before they have passed through the first service phase, receive the same (initial) reward $r(0)$ as jobs that are rejected upon arrival at the system. The difference lies in the holding costs ($h$ per period per job) and the consideration costs ($c$ per job), rejected jobs not imposing any holding and consideration costs on the system.

After the decision to continue or abort, one period of time elapses and we consider the next event.

**Service completion events** If the event is a real service completion event or an artificial service completion event concerning state $(i, N)$, $i \geq 1$, then we have the option to either continue or abort the service of the job under service. If the event is an artificial service completion event concerning state $(0, \cdot)$, then there is nothing to decide on at this point in time. In that case, again, we continue service per definition. As regards jobs in the queue, the same remarks can be made as in the case of an arrival event.

Again, after the decision to continue or abort, one period of time elapses and we consider the next event.
**Deliberate idling** Continuation of service in state \((i, N)\), \(i \geq 1\), can be seen as a (self-determined) decision to 'go to sleep', meaning that the server takes a one period vacation, while all work is left as it is. At the end of the period, either a new job arrives or there is an artificial service completion. We would like to note that, given the regularity assumption on the reward function, it can be shown that 'sleeping' is never optimal in terms of expected (discounted) profit, provided that \(\alpha r(N) + h \geq 0\), which is a very weak condition.

The form of this condition can be explained as follows. Sleeping can be advantageous if, and only if, the job currently under service has just entered or already resides in node \(N\). If the discount rate \(\alpha\) is strictly positive and all rewards are strictly negative, then we can let the actual reward of the job under service approach zero by going to sleep and letting the discounting do the work for us. If the holding costs are sufficiently low, then it might even be optimal to do so.

### 2.2 Mathematical model formulation

In this section, we summarize and complete the model in terms of a mathematical formulation. After that, we successively state and prove the main theorem.

Recapitulating, let \(i\) denote the number of jobs in the system and \(k\) the node the job under service resides in. Let \((i, k)\) be the state of the system for \(i \geq 1\) and \(0 \leq k \leq N\). If the system is empty, then the state is \((0, \cdot)\). Whenever \((0, k)\) appears in an expression, \((0, \cdot)\) can be substituted.

Let \(W_n(i, k)\) denote the maximum expected \(n\)-period \(\alpha\)-discounted profit when the current state, just before the next decision to continue or abort, is \((i, k)\). At the start of the process, the system is assumed to be in such a state. Note that the points in time considered here coincide with certain ones just after an arrival or service completion event and the corresponding decision to accept or reject if it was an arrival event.

Let \(W_n(i, k, \pi)\) denote the maximum expected \(n\)-period \(\alpha\)-discounted profit when the current state, just before the next decision to continue or abort, is \((i, k)\), and given that decision \(\pi\) is chosen in that state, where \(\pi \in \{\text{continue, abort}\}\). Let \(\pi^*\) denote the optimal decision, so \(W_n(i, k) = W_n(i, k, \pi^*)\).

Furthermore, let \(W_n(i, k, \text{arr})\) denote the maximum expected \(n\)-period \(\alpha\)-discounted profit when the current state is \((i, k)\), given that at this very point in time an arrival event occurs. Analogously, let \(W_n(i, k, \text{arr}, \pi)\) denote the maximum expected \(n\)-period \(\alpha\)-discounted profit when the current state is \((i, k)\), given that at this very point in time an arrival event occurs, and given that decision \(\pi\) is chosen in that state, where \(\pi \in \{\text{accept, reject}\}\). Again, let \(\pi^*\) denote the optimal decision, so \(W_n(i, k, \text{arr}) = W_n(i, k, \text{arr}, \pi^*)\).

Finally, when time hits zero, all jobs residing in the queue yield their initial reward \(r(0)\), while the job under service, residing in node \(k\) for some \(0 \leq k \leq N\), yields its current cashable reward, which is \(r(k)\).

Then our model is defined by the following Dynamic Programming Equations (DPEs). To save space, we will usually write ab for abort and co for continue in formal expressions (and also ac for accept and rj for reject).
For $n \geq 1$:

$$W_n(0, \cdot) = W_n(0, \cdot, \text{co})$$

$$W_n(i, k) = \max \{W_n(i, k, \text{co}), W_n(i, k, \text{ab})\} \quad i \geq 1 \quad 0 \leq k \leq N$$

where

$$W_n(0, \cdot, \text{co}) = \lambda W_{n-1}(0, \cdot, \text{arr}) + \mu W_{n-1}(0, \cdot)$$

$$W_n(i, k, \text{co}) = \lambda W_{n-1}(i, k, \text{arr}) + \mu W_{n-1}(i, k+1) - ih \quad i \geq 1 \quad 0 \leq k < N$$

$$W_n(i, N, \text{co}) = \lambda W_{n-1}(i, N, \text{arr}) + \mu W_{n-1}(i, N) - ih \quad i \geq 1$$

$$W_n(i, k, \text{ab}) = r(k) + W_n(i-1, 0, \text{co}) \quad i \geq 1 \quad 0 \leq k \leq N$$

and

$$W_{n-1}(0, \cdot, \text{arr}) = \max \{W_{n-1}(1, 0) - c, W_{n-1}(0, \cdot) + r(0)\}$$

$$W_{n-1}(i, k, \text{arr}) = \max \{W_{n-1}(i+1, k) - c, W_{n-1}(i, k) + r(0)\} \quad i \geq 1 \quad 0 \leq k \leq N$$

and finally

$$W_0(0, \cdot) = 0$$

$$W_0(i, k) = r(k) + (i-1)r(0) \quad i \geq 1 \quad 0 \leq k \leq N$$

Remark 2

The assumption that the reward function is non-decreasing, is essentially an implication of the assumption that the reward function is concave. If $r(k)$ is not non-decreasing in $k$, but indeed concave, then there exists a unique node $k_z$, $0 \leq k_z < N$, for which

$$\forall_{0 \leq k \leq k_z} \quad r(k) \leq r(k_z),$$

$$\forall_{k_z < k \leq N} \quad r(k) < r(k_z).$$

It is evident, and straightforward to prove, that the optimal decision in states $(i, k), i \geq 1$ and $k_z \leq k \leq N$, is to abort service (provided that $\omega(N) + h \geq 0$, cf the paragraph on deliberate idling). Therefore, essentially, the maximum number of service phases $N$ and the reward function $r(k), 0 \leq k \leq N$, can be reduced to $k_z$ and $r(k), 0 \leq k \leq k_z$, respectively. This yields a non-decreasing reward function after all.
3 Overview of the results

We will prove the following theorem.

Theorem 1
{CHARACTERIZATION OF THE OPTIMAL ADMISSION/TERMINATION POLICY}
Let the remaining number of periods be $n$. Then the optimal admission/termination policy can be characterized as follows:

1. If it is optimal to reject an arriving job in state $(i, k)$, then it is optimal as well to reject it in all states $(j, l)$ with $j > i$, and in all states $(i, l)$ with $l < k$.

2. If it is optimal to abort the service of a job in state $(i, k)$, then it is optimal as well to abort service in all states $(j, l)$ with $j \geq i$ and $l \geq k$.

3. If it is optimal to accept an arriving job in state $(i, k)$, then it is optimal as well to continue the service of a job in all states $(j, 0)$ with $j \leq i$.

4. Let there be no consideration costs. If it is optimal to reject an arriving job in state $(i, k)$, then it is optimal as well to abort the service of a job in all states $(j, k)$ with $j > i$.

The result is intuitive but the proof becomes rather involved. A graphical representation of the structure of typical admission and termination policies is given below. The solid dots represent nodes for which jobs are rejected or terminated, respectively, and the two solid polylines capture the respective rejection and termination regions.

![Figure 1: Characterization of the optimal admission (lhs) and termination (rhs) policies](image)

We note that in the optimal admission policy either there is a unique (threshold) node $(i_0, k_0)$ where the polyline moves up one level, or the polyline consists of a single horizontal line at some job level $i_0$. In the optimal termination policy, the termination region does not cover nodes $(i, 0)$, $i \leq i_0$ or $i < i_0$, respectively. The termination region is further determined by a non-increasing step-function such that for all $i$ there exists a threshold node $k$ of $i$ and for all $k$ there exists a threshold level $i$ of $k$.

If there are no consideration costs, then there is a stronger reciprocity between the optimal admission and termination policies, as shown below.
Figure 2: Characterization of the optimal admission and termination policies under $c = 0$

Again, in the optimal admission policy either there is a unique node $(i_0, k_0)$ where the polyline moves up one level, or the polyline consists of a single horizontal line at some job level $i_0$. In the optimal termination policy, the termination region above job level $i_0$ is fully determined by the rejection region, as indicated by the dotted line in the figure. The termination region at and below job level $i_0$ is determined by a non-increasing step-function such that for all $i$ there exists a threshold node $k$ of $i$ and for all $k$ there exists a threshold level $i$ of $k$. Here, again, it is guaranteed that nodes $(i, 0), i < i_0$, are not covered.

4 The line of proof

The main technique to prove Theorem 1 will be to use induction on the remaining number of periods. In order to establish the theorem, we will prove the following monotonicity results.

Proposition 1  
{\textit{\{Key Proposition\}}}

Let $r(k)$ be non-decreasing and concave in $k$. Then, for all $n \geq 0$ and $i \geq 0$,

\begin{align*}
W_n(i+1, l) - W_n(i, l) & \geq W_n(i+2, k) - W_n(i+1, k), \\
W_n(i+1, k+1) - W_n(i, k+1) & \geq W_n(i+1, k) - W_n(i, k), \\
W_n(i+1, 1, \text{arr}) - W_n(i, 1, \text{arr}) & \geq W_n(i+2, k, \text{arr}) - W_n(i+1, k, \text{arr}), \\
W_n(i+1, k+1, \text{arr}) - W_n(i, k+1, \text{arr}) & \geq W_n(i+1, k, \text{arr}) - W_n(i, k, \text{arr}),
\end{align*}

\begin{align*}
W_{n+1}(i+1, l, \text{co}) - W_{n+1}(i, l, \text{co}) & \geq W_{n+1}(i+2, k, \text{co}) - W_{n+1}(i+1, k, \text{co}), \\
W_{n+1}(i+1, k+1, \text{co}) - W_{n+1}(i, k+1, \text{co}) & \geq W_{n+1}(i+1, k, \text{co}) - W_{n+1}(i, k, \text{co}),
\end{align*}

where (1), (3) and (5) hold for all $0 \leq k \leq N$ and $0 \leq l \leq N$, and (2), (4) and (6) for all $0 \leq k < N$.

In words, inequality (1) states that the value of an additional job in state $(i+1, k)$ does not surpass the value of an additional job in state $(i, l)$. Analogously, inequality (2) states that the value of an additional job in state $(i, k)$ does not surpass the value of an additional job in state $(i, k+1)$. 

8
It can easily be verified that the first two inequalities hold by definition for \( n = 0 \). The proof is further organized as follows. Assuming that (1) and (2) hold for some \( n \geq 0 \) (Step 0), we prove that (3) and (4) hold for \( n \) (Step 1). Using this result, we prove that (5) and (6) hold for \( n + 1 \) (Step 2). Finally, we prove that (1) and (2) also hold for \( n + 1 \) (Step 3). To establish this, we make use of two additional lemmas.

**Remark 3**

*Concavity of \( W_n(i, k) \)*

Inequality (1) implies

\[
W_n(i + 1, k) - W_n(i, k) \geq W_n(i + 2, k) - W_n(i + 1, k)
\]

for all \( n \geq 0 \), \( i \geq 0 \) and \( 0 \leq k \leq N \). Thus, \( W_n(i, k) \) is concave in \( i \) for all fixed combinations of \( n \geq 0 \) and \( 0 \leq k \leq N \).

## 5 Proof of the Key Proposition

Throughout this section Fundamental Assumption 1 is in effect.

**Lemma 1**

Let either \( s_m = (i_m, k_m) \) or \( s_m = (i_m, k_m, \alpha) \), \( m = 1, \ldots, 4 \), and let \( \phi \) and \( \psi \) be authorized decisions, given \( s_m \). Then

\[
W_n(s_1, \phi) - W_n(s_2, \pi^*) \geq W_n(s_3, \pi^*) - W_n(s_4, \psi)
\]

implies

\[
W_n(s_1, \pi^*) - W_n(s_2, \pi^*) \geq W_n(s_3, \pi^*) - W_n(s_4, \pi^*).
\]

**Proof.** Assume (7) for certain decisions \( \phi \) and \( \psi \). By definition, \( W_n(s_1, \pi^*) \geq W_n(s_1, \phi) \) and \( W_n(s_4, \pi^*) \geq W_n(s_4, \psi) \). Therefore,

\[
W_n(s_1, \pi^*) - W_n(s_2, \pi^*) \geq W_n(s_1, \phi) - W_n(s_2, \pi^*)
\]

\[
\geq W_n(s_3, \pi^*) - W_n(s_4, \psi)
\]

\[
\geq W_n(s_3, \pi^*) - W_n(s_4, \pi^*).
\]

\( \square \)

We will use Lemma 1 in the following way. When distinguishing between all possible combinations of optimal decisions in certain states \( S_2 \) and \( S_3 \), we choose \( \phi \) and \( \psi \) such that (7) holds.

**Proof of the Key Proposition. Step 0.** We verify that (1) and (2) hold for \( n = 0 \).

Let \( i = 0 \). Then, for \( 0 \leq k \leq N \) and \( 0 \leq l \leq N \),

\[
W_0(1, l) - W_0(0, \cdot) = r(l) - 0 = r(l)
\]

\[
\geq \{ r(k) \text{ non-decreasing in } k \}
\]

\[
r(0)
\]

\[
= [r(k) + r(0)] - r(k)
\]

\[
= W_0(2, k) - W_0(1, k)
\]
and, for \(0 \leq k < N\),
\[
W_0(1,k + 1) - W_0(0, \cdot) = r(k + 1) - 0 = r(k + 1) \\
\geq \{r(k) \text{ non-decreasing in } k\} \\
r(k) \\
= W_0(1,k) - W_0(0, \cdot).
\]

Next, let \(i \geq 1\). Then, for \(0 \leq k \leq N\) and \(0 \leq l \leq N\),
\[
W_0(i + 1, l) - W_0(i, l) = r(l) + ir(0) - [r(l) + (i - 1)r(0)] \\
= r(0) \\
= r(k) + (i + 1)r(0) - [r(k) + ir(0)] \\
= W_0(i + 2, k) - W_0(i + 1, k)
\]

and, for \(0 \leq k < N\),
\[
W_0(i + 1, k + 1) - W_0(i, k + 1) = r(k + 1) + ir(0) - [r(k + 1) + (i - 1)r(0)] \\
= r(0) \\
= r(k) + ir(0) - [r(k) + (i - 1)r(0)] \\
= W_0(i + 1, k) - W_0(i, k).
\]

Thus, inequalities (1) and (2) hold for \(n = 0\).

Assume that for some \(n \geq 0\), inequality (1) holds for all \(i \geq 0\), \(0 \leq k \leq N\) and \(0 \leq l \leq N\), and (2) for all \(i \geq 0\) and \(0 \leq k < N\). This will be our induction hypothesis.

**Step 1.** Under the induction hypothesis, we show that (3) and (4) hold for \(n\).

We will first prove (3). Let \(i \geq 0\), \(0 \leq k \leq N\) and \(0 \leq l \leq N\).

The next decision, say \(d_1\), prescribed by the (optimal) policy corresponding to \(W_n(i, l, \text{arr})\), is either to accept or to reject the new job. Clearly, this also holds for the next decision, say \(d_2\), prescribed by the (optimal) policy corresponding to \(W_n(i + 2, k, \text{arr})\). There are at most four joint cases \((d_1, d_2)\). We will show that inequality (3) holds for each case, irrespective of the question whether that case will actually occur (cf Remark 4).

The four cases can be presented as follows, where \(\mathcal{A}\) indicates that accept is optimal and \(\mathcal{R}\) indicates that accept is not optimal:

| \(\mathcal{A}\) | \(W_n(i + 1, l) - c \geq W_n(i, l) + r(0) \land W_n(i + 3, k) - c \geq W_n(i + 2, k) + r(0)\) |
| \(\mathcal{A}\mathcal{R}\) | \(W_n(i + 1, l) - c \geq W_n(i, l) + r(0) \land W_n(i + 3, k) - c < W_n(i + 2, k) + r(0)\) |
| \(\mathcal{R}\mathcal{A}\) | \(W_n(i + 1, l) - c < W_n(i, l) + r(0) \land W_n(i + 3, k) - c \geq W_n(i + 2, k) + r(0)\) |
| \(\mathcal{R}\mathcal{R}\) | \(W_n(i + 1, l) - c < W_n(i, l) + r(0) \land W_n(i + 3, k) - c < W_n(i + 2, k) + r(0)\) |

Then,
- under $\mathcal{AA}$,

\[
W_n(i + 1, l, arr, ac) - W_n(i, l, arr) = W_n(i + 2, l) - c - [W_n(i + 1, l) - c] \\
= W_n(i + 2, l) - W_n(i + 1, l) \\
\geq \text{induction hypothesis; (1)} \\
W_n(i + 3, k) - W_n(i + 2, k) \\
= W_n(i + 3, k) - c - [W_n(i + 2, k) - c] \\
= W_n(i + 2, k, arr) - W_n(i + 1, k, arr, ac);
\] (8)

- under $\mathcal{AR}$,

\[
W_n(i + 1, l, arr, x j) - W_n(i, l) = W_n(i + 1, l) + r(0) - [W_n(i, l) + r(0)] \\
= W_n(i + 1, l) - W_n(i, l) \\
\geq \text{induction hypothesis; (1)} \\
W_n(i + 2, k) - W_n(i + 1, k) \\
\geq \text{induction hypothesis; (1)} \\
W_n(i + 3, k) - W_n(i + 2, k) \\
= W_n(i + 3, k) - c - [W_n(i + 2, k) - c] \\
= W_n(i + 2, k, arr) - W_n(i + 1, k, arr, ac); \] (9)

- under $\mathcal{RA}$,

\[
W_n(i + 1, l, arr, x j) - W_n(i, l, arr) = W_n(i + 1, l) + r(0) - [W_n(i, l) + r(0)] \\
= W_n(i + 1, l) - W_n(i, l) \\
\geq \text{induction hypothesis; (1)} \\
W_n(i + 2, k) - W_n(i + 1, k) \\
= W_n(i + 2, k) + r(0) - [W_n(i + 1, k) + r(0)] \\
= W_n(i + 2, k, arr) - W_n(i + 1, k, arr, x j). \] (10)

- under $\mathcal{RR}$,

\[
W_n(i + 1, l, arr, x j) - W_n(i, l) = W_n(i + 1, l) + r(0) - [W_n(i, l) + r(0)] \\
= W_n(i + 1, l) - W_n(i, l) \\
\geq \text{induction hypothesis; (1)} \\
W_n(i + 2, k) - W_n(i + 1, k) \\
= W_n(i + 2, k) + r(0) - [W_n(i + 1, k) + r(0)] \\
= W_n(i + 2, k, arr) - W_n(i + 1, k, arr, x j). \] (11)

Finally, apply Lemma 1 to each of the relations (8) through (11) to obtain (3). This ends our proof of (3).

We now shift our attention to (4). The proof resembles the one of (3). Let $i \geq 0$ and $0 \leq k < N$. Again, we distinguish four cases:

\begin{align*}
\mathcal{AA} & : W_n(i + 1, k + 1) - c \geq W_n(i, k + 1) + r(0) \wedge W_n(i + 2, k) - c \geq W_n(i + 1, k) + r(0), \\
\mathcal{AR} & : W_n(i + 1, k + 1) - c \geq W_n(i, k + 1) + r(0) \wedge W_n(i + 2, k) - c < W_n(i + 1, k) + r(0), \\
\mathcal{RA} & : W_n(i + 1, k + 1) - c < W_n(i, k + 1) + r(0) \wedge W_n(i + 2, k) - c \geq W_n(i + 1, k) + r(0), \\
\mathcal{RR} & : W_n(i + 1, k + 1) - c < W_n(i, k + 1) + r(0) \wedge W_n(i + 2, k) - c < W_n(i + 1, k) + r(0).
\end{align*}
Then,
- under $AA$,
\[
W_n(i + 1, k + 1, \text{arr, } ac) - W_n(i, k + 1, \text{arr}) = W_n(i + 2, k + 1) - c - [W_n(i + 1, k + 1) - c] = W_n(i + 2, k + 1) - W_n(i + 1, k + 1) \geq \text{induction hypothesis; (2)} \]
\[
W_n(i + 2, k) - W_n(i + 1, k) = W_n(i + 2, k) - c - [W_n(i + 1, k) - c] = W_n(i + 1, k, \text{arr}) - W_n(i, k, \text{arr, } ac); \quad (12)
\]
- under $A\mathcal{R}$,
\[
W_n(i + 1, k + 1, \text{arr, } xj) - W_n(i, k + 1, \text{arr}) = W_n(i + 1, k + 1) + r(0) - [W_n(i + 1, k + 1) - c] = r(0) - c = W_n(i + 1, k) + r(0) - [W_n(i + 1, k) - c] = W_n(i + 1, k, \text{arr}) - W_n(i, k, \text{arr, } ac); \quad (13)
\]
- under $R.A$,
\[
W_n(i + 1, k + 1, \text{arr, } xj) - W_n(i, k + 1, \text{arr}) = W_n(i + 1, k + 1) + r(0) - [W_n(i, k + 1) + r(0)] = W_n(i + 1, k + 1) - W_n(i, k + 1) \geq \text{induction hypothesis; (1)} \]
\[
W_n(i + 2, k + 1) - W_n(i + 1, k + 1) \geq \text{induction hypothesis; (2)} \]
\[
W_n(i + 2, k) - W_n(i + 1, k) = W_n(i + 2, k) - c - [W_n(i + 1, k) - c] = W_n(i + 1, k, \text{arr}) - W_n(i, k, \text{arr, } ac); \quad (14)
\]
- under $R\mathcal{R}$,
\[
W_n(i + 1, k + 1, \text{arr, } xj) - W_n(i, k + 1, \text{arr}) = W_n(i + 1, k + 1) + r(0) - [W_n(i, k + 1) + r(0)] = W_n(i + 1, k + 1) - W_n(i, k + 1) \geq \text{induction hypothesis; (2)} \]
\[
W_n(i + 1, k) - W_n(i, k) = W_n(i + 1, k) + r(0) - [W_n(i, k) + r(0)] = W_n(i + 1, k, \text{arr}) - W_n(i, k, \text{arr, } xj). \quad (15)
\]

Finally, apply Lemma 1 to each of the relations (12) through (15) to obtain (4). This ends our proof of (4).
Remark 4
Case RA in the proof of (3) constitutes a violation of the Key Proposition, because under RA,
\[ r(0) + c > \{ RA \text{ assumed} \} \]
\[ W_n(i + 1, l) - W_n(i, l) \]
\[ \geq \{ \text{induction hypothesis; (1)} \} \]
\[ W_n(i + 2, k) - W_n(i + 1, k) \]
\[ \geq \{ \text{induction hypothesis; (1)} \} \]
\[ W_n(i + 3, k) - W_n(i + 2, k) \]
\[ \geq \{ RA \text{ assumed} \} \]
\[ r(0) + c, \]
which is clearly a contradiction. Given the correctness of the Key Proposition, this proves that case RA in the proof of (3) is an impossible joint case.

Step 2. Assuming (1) through (4), we show that (5) and (6) hold for \( n + 1 \). For this purpose, we define \( m^- := \min\{m, N\} \) for all integers \( m \geq 0 \).

Let \( i \geq 0 \). Then, for \( 0 \leq k \leq N \) and \( 0 \leq l \leq N \),
\[ W_{n+1}(i + 1, l, co) - W_{n+1}(i, l, co) = \lambda[W_n(i + 1, l, arr) - W_n(i, l, arr)] + \mu[W_n(i + 1, (l + 1)^-) - W_n(i, (l + 1)^-)] - h \]
\[ \geq \{ \text{induction hypothesis; (3); (1)} \} \]
\[ \lambda[W_n(i + 2, k, arr) - W_n(i + 1, k, arr)] + \mu[W_n(i + 2, (k + 1)^-) - W_n(i + 1, (k + 1)^-)] - h \]
\[ = W_{n+1}(i + 2, k, co) - W_{n+1}(i + 1, k, co) \]
and, for \( 0 \leq k < N \),
\[ W_{n+1}(i + 1, k + 1, co) - W_{n+1}(i, k + 1, co) = \lambda[W_n(i + 1, k + 1, arr) - W_n(i, k + 1, arr)] + \mu[W_n(i + 1, (k + 2)^-) - W_n(i, (k + 2)^-)] - h \]
\[ \geq \{ \text{induction hypothesis; (4); (2)} \} \]
\[ = \lambda[W_n(i + 1, k, arr) - W_n(i, k, arr)] + \mu[W_n(i + 1, (k + 1)^-) - W_n(i, (k + 1)^-)] - h \]
\[ = W_{n+1}(i + 1, k, co) - W_{n+1}(i, k, co). \]

In order to perform the third step of our proof, we need the following lemma, which exploits the concavity of \( r(k) \).

Lemma 2
For all \( n \geq 0, i \geq 0 \) and \( 0 \leq k < N \),
\[ W_{n+1}(i, k + 1, co) - W_{n+1}(i, k, co) \leq r(k + 1) - r(k). \] (16)
Proof. The inequality holds by definition for \( i = 0 \). Using the same technique as in the proof of the Key Proposition, it can be proven in a straightforward manner, by induction, that for all \( n \geq 0, i \geq 0 \) and \( 0 \leq k < N \),

\[
\begin{align*}
W_n(i, k + 1) - W_n(i, k) \\
W_{n+1}(i, k + 1, \text{co}) - W_{n+1}(i, k, \text{co}) \\
W_n(i, k + 1, \text{arr}) - W_n(i, k, \text{arr})
\end{align*}
\] \( \leq r(k + 1) - r(k) \).

We choose not to reproduce this proof here. Instead, we give a sample path argument.

Consider two \( n \)-period process instances of our model, one starting in \((i, k + 1)\), for some \( i \geq 1 \) and \( 0 \leq k < N \), with the extension that decision continue is chosen (instance \( I_1 \)), and one starting in \((i, k)\), also with the extension that decision continue is chosen (instance \( I_0 \)). We couple all events (all arrival and service completion events and, if applicable, the event that the process vanishes) and all decisions (accept versus reject and continue versus abort). To be precise, once both processes have carried through the initial continue operation, instance \( I_0 \) copies the optimal decisions taken in instance \( I_1 \). This is feasible, because \( I_0 \) and \( I_1 \) feature the same number of jobs in the system and the same remaining number of periods.

Then, for both processes, the costs of continuing service are identical, and so are the costs and rewards resulting from the admission and rejection, respectively, of new jobs while continuing the service of the job currently under service. If the processes vanish prior to time zero and prior to the termination of any jobs, then the difference in profit amounts to zero, which is at most \( r(k + 1) - r(k) \) for any \( 0 \leq k < N \). If, on the other hand, the processes do not vanish before time hits zero, then, at some point in time, the job under service is terminated. This leaves us with job rewards \( r((l + 1)^-) \) and \( r(l^-) \), for \( I_1 \) and \( I_0 \) respectively, for some \( k \leq l \leq N \). Here, \( r((l + 1)^-) - r(l^-) \leq r(k + 1) - r(k) \), because of the concavity of \( r(k) \). After this abort operation, both processes become and remain identical. If the job under service is terminated exactly at time zero, then the additional profits from queued jobs are equal for both processes, because they feature the same number of queued jobs.

\[ \Box \]

Step 3. Assuming (1) through (6), we show that (1) and (2) hold for \( n + 1 \) as well. The proofs are similar to the ones of (3) and (4).

We will first prove that (1) holds for \( n + 1 \). Let \( i \geq 0, 0 \leq k \leq N \) and \( 0 \leq l \leq N \).

The next decision, say \( d_1 \), prescribed by the (optimal) policy corresponding to \( W_{n+1}(i, l) \), is either to continue or to abort the job under service. Clearly, this also holds for the next decision, say \( d_2 \), prescribed by the (optimal) policy corresponding to \( W_{n+1}(i + 2, k) \). There are at most four joint cases \((d_1, d_2)\). We will show that inequality (1) holds for each case.

The four cases can be presented as follows, where \( C \) indicates that continue is optimal and \( A \) indicates that continue is not optimal:

\[
\begin{align*}
\text{CC} : & \quad W_{n+1}(i, l, \text{co}) \geq W_{n+1}(i, l, \text{ab}) \land W_{n+1}(i + 2, k, \text{co}) \geq W_{n+1}(i + 2, k, \text{ab}), \\
\text{CA} : & \quad W_{n+1}(i, l, \text{co}) \geq W_{n+1}(i, l, \text{ab}) \land W_{n+1}(i + 2, k, \text{co}) < W_{n+1}(i + 2, k, \text{ab}), \\
\text{AC} : & \quad W_{n+1}(i, l, \text{co}) < W_{n+1}(i, l, \text{ab}) \land W_{n+1}(i + 2, k, \text{co}) \geq W_{n+1}(i + 2, k, \text{ab}), \\
\text{AA} : & \quad W_{n+1}(i, l, \text{co}) < W_{n+1}(i, l, \text{ab}) \land W_{n+1}(i + 2, k, \text{co}) < W_{n+1}(i + 2, k, \text{ab}).
\end{align*}
\]

The cases \( \text{AC} \) and \( \text{AA} \) vanish for \( i = 0 \), because abort is not an option in state \((0, \cdot)\).
Then,

- under $CC$,

$$W_{n+1}(i+1,l,co) - W_{n+1}(i,l) = W_{n+1}(i+1,l,co) - W_{n+1}(i,l,co)$$
$$\geq \{\text{induction hypothesis; (5)}\}$$
$$W_{n+1}(i+2,k,co) - W_{n+1}(i+1,k,co)$$
$$= W_{n+1}(i+2,k) - W_{n+1}(i+1,k,co); \quad (17)$$

- under $CA$,

$$W_{n+1}(i+1,l,co) - W_{n+1}(i,l) = W_{n+1}(i+1,l,co) - W_{n+1}(i,l,co)$$
$$\geq \{\text{induction hypothesis; (6)}\}$$
$$W_{n+1}(i+1,0,co) - W_{n+1}(i,0,co)$$
$$= r(k) + W_{n+1}(i+1,0,co) - [r(k) + W_{n+1}(i,0,co)]$$
$$= W_{n+1}(i+2,k,ab) - W_{n+1}(i+1,k,ab)$$
$$= W_{n+1}(i+2,k) - W_{n+1}(i+1,k,ab); \quad (18)$$

- under $AC$,

$$W_{n+1}(i+1,l,ab) - W_{n+1}(i,l) = W_{n+1}(i+1,l,ab) - W_{n+1}(i,l,ab)$$
$$= r(l) + W_{n+1}(i,0,co) - [r(l) + W_{n+1}(i-1,0,co)]$$
$$= W_{n+1}(i,0,co) - W_{n+1}(i-1,0,co)$$
$$\geq \{\text{induction hypothesis; (5)}\}$$
$$W_{n+1}(i+1,k,co) - W_{n+1}(i,k,co)$$
$$\geq \{\text{induction hypothesis; (5)}\}$$
$$W_{n+1}(i+2,k,co) - W_{n+1}(i+1,k,co)$$
$$= W_{n+1}(i+2,k) - W_{n+1}(i+1,k,co); \quad (19)$$

- under $AA$,

$$W_{n+1}(i+1,l,ab) - W_{n+1}(i,l) = W_{n+1}(i+1,l,ab) - W_{n+1}(i,l,ab)$$
$$= r(l) + W_{n+1}(i,0,co) - [r(l) + W_{n+1}(i-1,0,co)]$$
$$= W_{n+1}(i,0,co) - W_{n+1}(i-1,0,co)$$
$$\geq \{\text{induction hypothesis; (5)}\}$$
$$W_{n+1}(i+1,0,co) - W_{n+1}(i,0,co)$$
$$= r(k) + W_{n+1}(i+1,0,co) - [r(k) + W_{n+1}(i,0,co)]$$
$$= W_{n+1}(i+2,k,ab) - W_{n+1}(i+1,k,ab)$$
$$= W_{n+1}(i+2,k) - W_{n+1}(i+1,k,ab). \quad (20)$$

Finally, apply Lemma 1 to each of the relations (17) through (20) to obtain (1) for $n+1$. This ends our proof of (1) for $n+1$. 

15
We now shift our attention to (2) and prove that this inequality holds for $n+1$. The proof resembles the one of (1) for $n+1$. Let $i \geq 0$ and $0 \leq k < N$. Again, we distinguish four cases:

- **CC**: $W_{n+1}(i, k+1, \text{co}) \geq W_{n+1}(i, k+1, \text{ab}) \land W_{n+1}(i+1, k, \text{co}) \geq W_{n+1}(i+1, k, \text{ab})$
- **CA**: $W_{n+1}(i, k+1, \text{co}) \geq W_{n+1}(i, k+1, \text{ab}) \land W_{n+1}(i+1, k, \text{co}) < W_{n+1}(i+1, k, \text{ab})$
- **AC**: $W_{n+1}(i, k+1, \text{co}) < W_{n+1}(i, k+1, \text{ab}) \land W_{n+1}(i+1, k, \text{co}) \geq W_{n+1}(i+1, k, \text{ab})$
- **AA**: $W_{n+1}(i, k+1, \text{co}) < W_{n+1}(i, k+1, \text{ab}) \land W_{n+1}(i+1, k, \text{co}) < W_{n+1}(i+1, k, \text{ab})$

Again, the cases AC and AA vanish for $i = 0$ by definition.

Then,

- under CC,
  \[ W_{n+1}(i+1, k+1, \text{co}) - W_{n+1}(i, k+1) = W_{n+1}(i+1, k+1, \text{co}) - W_{n+1}(i, k+1) \geq \text{induction hypothesis; (6)} \]
  \[ = W_{n+1}(i+1, k, \text{co}) - W_{n+1}(i, k, \text{co}) \]
  \[ = W_{n+1}(i+1, k) - W_{n+1}(i, k, \text{co}); \quad (21) \]

- under CA,
  \[ W_{n+1}(i+1, k+1, \text{ab}) - W_{n+1}(i, k+1) = W_{n+1}(i+1, k+1, \text{ab}) - W_{n+1}(i, k+1) \geq \text{Lemma 2} \]
  \[ = r(k+1) + W_{n+1}(i, 0, \text{co}) - W_{n+1}(i, k+1, \text{co}) \]
  \[ = W_{n+1}(i+1, k, \text{ab}) - W_{n+1}(i, k, \text{co}) \]
  \[ = W_{n+1}(i+1, k) - W_{n+1}(i, k, \text{co}); \quad (22) \]

- under AC,
  \[ W_{n+1}(i+1, k+1, \text{ab}) - W_{n+1}(i, k+1) = W_{n+1}(i+1, k+1, \text{ab}) - W_{n+1}(i, k+1) \geq \text{induction hypothesis; (5)} \]
  \[ = W_{n+1}(i+1, k, \text{co}) - W_{n+1}(i, k, \text{co}) \]
  \[ = W_{n+1}(i+1, k) - W_{n+1}(i, k, \text{co}); \quad (23) \]

- under AA,
  \[ W_{n+1}(i+1, k+1, \text{ab}) - W_{n+1}(i, k+1) = W_{n+1}(i+1, k+1, \text{ab}) - W_{n+1}(i, k+1, \text{ab}) \geq \text{Lemma 2} \]
  \[ = r(k+1) + W_{n+1}(i, 0, \text{co}) - r(k+1) + W_{n+1}(i-1, 0, \text{co}) \]
  \[ = W_{n+1}(i, 0, \text{co}) - W_{n+1}(i-1, 0, \text{co}) \]
  \[ = W_{n+1}(i+1, k, \text{ab}) - W_{n+1}(i, k, \text{ab}) \]
  \[ = W_{n+1}(i+1, k) - W_{n+1}(i, k, \text{ab}); \quad (24) \]

Finally, apply Lemma 1 to each of the relations (21) through (24) to obtain (2) for $n+1$. This ends our proof of (2) for $n+1$.

\[ \square \]

This concludes our proof of the Key Proposition.

\[ \square \]
Remark 5

Concavity of $r(k)$ is only required at one particular point in the entire proof of the Key Proposition, namely in the derivation of (22), where Lemma 2 is used.

We now derive Theorem 1 from the Key Proposition by means of a number of corollaries.

Corollary 1

Let $n \geq 0$, $i \geq 0$ and $0 \leq k \leq N$. If it is optimal to reject an arriving job in state $(i, k)$, then it is optimal to reject it in all states $(j, l)$ with $j > i$ and $0 \leq l \leq N$.

Proof. Let $n \geq 0$, $i \geq 0$ and $0 \leq k \leq N$, where $k = 0$ if $i = 0$. If suffices to show that

$$W_n(i, k, \text{arr}) = W_n(i, k) + r(0) \geq W_n(i + 1, k) - c$$

implies

$$W_n(i + 1, l, \text{arr}) = W_n(i + 1, l) + r(0) \geq W_n(i + 2, l) - c$$

for all $0 \leq l \leq N$.

Let

$$W_n(i, k) + r(0) \geq W_n(i + 1, k) - c,$$

so

$$W_n(i + 1, k) - W_n(i, k) \leq r(0) + c.$$

Then, for $0 \leq l \leq N$, by (1),

$$W_n(i + 2, l) - W_n(i + 1, l) \leq W_n(i + 1, k) - W_n(i, k) \leq r(0) + c,$$

so

$$W_n(i + 1, l) + r(0) \geq W_n(i + 2, l) - c.$$

□

Corollary 2

Let $n \geq 0$, $i \geq 0$ and $1 \leq k \leq N$. If it is optimal to reject an arriving job in state $(i, k)$, then it is optimal to reject it in all states $(i, l)$ with $0 \leq l < k$.

Proof. Corollary 2 holds by definition for $i = 0$. Let $n \geq 0$, $i \geq 1$ and $1 \leq k \leq N$. If suffices to show that

$$W_n(i, k, \text{arr}) = W_n(i, k) + r(0) \geq W_n(i + 1, k) - c$$

implies

$$W_n(i, k - 1, \text{arr}) = W_n(i, k - 1) + r(0) \geq W_n(i + 1, k - 1) - c.$$
Let
\[ W_n(i, k) + r(0) \geq W_n(i + 1, k) - c, \]
so
\[ W_n(i + 1, k) - W_n(i, k) \leq r(0) + c. \]
Then, by (2),
\[ W_n(i + 1, k - 1) - W_n(i, k - 1) \leq W_n(i + 1, k) - W_n(i, k) \leq r(0) + c, \]
so
\[ W_n(i, k - 1) + r(0) \geq W_n(i + 1, k - 1) - c. \]

The combination of Corollaries 1 and 2 is exactly part 1 of Theorem 1.

**Corollary 3**
Let \( n \geq 1, i \geq 1 \) and \( 0 \leq k \leq N \). If it is optimal to abort the service of a job in state \((i, k)\), then it is optimal to abort it in all states \((j, l)\) with \( j \geq i \) and \( k \leq l \leq N \).

**Proof.** Let \( n \geq 1, i \geq 1 \) and \( 0 \leq k \leq N \). It suffices to show that
\[ W_n(i, k, ab) \geq W_n(i, k, co) \]
implies
\[ W_n(i + q, l, ab) \geq W_n(i + q, l, co) \]
for all \( k \leq l \leq N \) and \( q \in \{0, 1\} \).

Suppose that \( W_n(i, k, ab) \geq W_n(i, k, co) \), but \( W_n(i + q, l, ab) < W_n(i + q, l, co) \) for some \( k \leq l \leq N \) and \( q \in \{0, 1\} \). Then
\[
\begin{align*}
W_n(i, k, ab) &= r(k) + W_n(i - 1, 0, co) \geq W_n(i, k, co), \\
-W_n(i + q, l, ab) &= -r(l) + W_n(i - 1 + q, 0, co) > -W_n(i + q, l, co).
\end{align*}
\]

Thus, for \( q = 0 \),
\[ W_n(i, l, co) - W_n(i, k, co) > r(l) - r(k), \]
which is clearly a contradiction for \( k = l \). For \( k < l \) it contradicts Lemma 2 (repeated application).

Alternatively, for \( q = 1 \),
\[ W_n(i - 1, 0, co) - W_n(i, 0, co) + r(k) - r(l) > W_n(i, k, co) - W_n(i + 1, l, co), \]
\[ W_n(i, 0, co) - W_n(i - 1, 0, co) < W_n(i + 1, l, co) - W_n(i, k, co) + r(k) - r(l) \]
\[ \leq \text{(Lemma 2 (repeated application))} \]
\[ W_n(i + 1, l, co) - W_n(i, k, co) + W_n(i, k, co) - W_n(i, l, co) = W_n(i + 1, l, co) - W_n(i, l, co), \]
contradicting inequality (5).

Corollary 3 is exactly part 2 of Theorem 1.

Corollary 4
Let \( n \geq 1, i \geq 1 \) and \( 0 \leq k \leq N \). If it is optimal to accept an arriving job in state \((i, k)\), then it is optimal to continue service in state \((i, 0)\).

Proof. Let \( n \geq 1, i \geq 1 \) and \( 0 \leq k \leq N \). It suffices to show that
\[ W_n(i, k, arr) = W_n(i + 1, k) - c \geq W_n(i, k) + r(0) \]
implies
\[ W_n(i, 0, co) \geq W_n(i, 0, ab). \]

Suppose that \( W_n(i, k, arr) = W_n(i + 1, k) - c \), but \( W_n(i, 0, co) < W_n(i, 0, ab) \). Then
\[ W_n(i + 1, k) - c \geq W_n(i, k) + r(0), \]
\[ -W_n(i, 0, co) > -W_n(i - 1, 0, co) - r(0), \]
so
\[ W_n(i + 1, k) - W_n(i, k) - c > W_n(i, 0, co) - W_n(i - 1, 0, co). \]

We distinguish between three joint cases with respect to the (optimal) decisions corresponding to \( W_n(i + 1, k) \) and \( W_n(i, k) \). The case \((co, ab)\) is an impossible joint case, according to Corollary 3. The remaining cases are:
1. \((co, co)\),
2. \((ab, co)\),
3. \((ab, ab)\).

Then,
- given case 1,
\[ W_n(i, 0, co) - W_n(i - 1, 0, co) \geq \text{(5)} \]
\[ W_n(i + 1, k, co) - W_n(i, k, co) \]
\[ = W_n(i + 1, k) - W_n(i, k); \]

(26)
- given case 2,
\[ W_n(i, k, co) \geq r(k) + W_n(i-1, 0, co), \]
so
\[ r(k) - W_n(i, k, co) \leq -W_n(i-1, 0, co), \]
and therefore
\[ W_n(i, 0, co) - W_n(i-1, 0, co) \geq \{27\} \]
\[ r(k) + W_n(i, 0, co) - W_n(i, k, co) \]
\[ = W_n(i+1, k) - W_n(i, k); \]
(28)
- given case 3,
\[ W_n(i, 0, co) - W_n(i-1, 0, co) = W_n(i, 0, co) + r(k) - [W_n(i-1, 0, co) + r(k)] \]
\[ = W_n(i+1, k) - W_n(i, k). \]
(29)
Finally, note that (26), (28) and (29) all imply
\[ W_n(i, 0, co) - W_n(i-1, 0, co) \geq W_n(i+1, k) - W_n(i, k) \geq W_n(i+1, k) - W_n(i, k) - c, \]
contradicting (25) in each case.

Together, Corollaries 3 and 4 constitute part 3 of Theorem 1.

**Corollary 5**

Let \( c = 0 \) and let \( n \geq 1, i \geq 0 \) and \( 0 \leq k \leq N \). If it is optimal to reject an arriving job in state \((i, k)\), then it is optimal to abort service in all states \((j, k)\) with \( j > i \).

**Proof.** Let \( c = 0 \) and let \( n \geq 1, i \geq 0 \) and \( 0 \leq k \leq N \). It suffices to show that the combination of
\[ W_n(i, k, arr) = W_n(i, k) + r(0) \geq W_n(i+1, k) \]
and
\[ W_n(i+1, k) = W_n(i+1, k, co) > W_n(i+1, k, ab) = W_n(i, 0, co) + r(k) \]
leads to a contradiction.

Assume (30) and (31). Consider the case that \( i = 0 \). Then
\[ W_n(0, \cdot) + r(0) + W_n(1, k) > \{30\}; \{31\} \]
\[ W_n(1, k) + W_n(0, \cdot, co) + r(k) \]
\[ = W_n(1, k) + W_n(0, \cdot) + r(k), \]
so \( r(0) > r(k) \), which is clearly a contradiction.
Alternatively, consider the case that \( i \geq 1 \). Then, by part 2 of Theorem 1,
\[
W_n(i, k) = W_n(i, k, \text{co}) \geq W_n(i, k, \text{ab}) = W_n(i - 1, 0, \text{co}) + r(k).
\]  
(32)

From this,
\[
W_n(i, k, \text{co}) + r(0) = \begin{cases} W_n(i, k) + r(0) \\ \geq \begin{cases} W_n(i + 1, k) \\ > \begin{cases} W_n(i, 0, \text{co}) + r(k), \end{cases} \end{cases} \end{cases}
\]

which is clearly a contradiction for \( k = 0 \). For \( k > 0 \) it contradicts Lemma 2 (repeated application).

Corollary 5 is exactly part 4 of Theorem 1. Summarizing, Corollaries 1 through 5 constitute Theorem 1.

6 Infinite time horizon

So far we only considered a finite time horizon, i.e., a finite number of periods. The extension of our results to the infinite time horizon cases (discounted and average) is not always trivial. In case the holding costs per job per period of time are strictly positive, one may show that the problem is essentially a finite state problem, because accepting a job when the system is very busy will lead to expected costs for that job that are higher than the immediate costs of rejection. Then the resulting Markov chain is recurrent (for each initial state the empty system is reached with probability 1) and therefore the threshold properties carry over to the infinite time horizon case.

More difficult are the many variants with \( h = 0 \). We will not consider all cases in detail. Instead, we make a few remarks about some of the possible situations. These will demonstrate the complexity. The interesting examples are those with \( \lambda N / \mu > 1 \), so when we can not treat all jobs completely. (If \( \lambda N / \mu < 1 \), and \( -c + r(N) > r(0) \), then we will accept all jobs and provide full service to each of the jobs.) There are differences between the average profit case and the discounted profit case.

**Average profit case** If \( \lambda N / \mu > 1 \), then the optimal (admission) strategy will prescribe to accept a certain fraction \( p \geq 0 \) of all jobs. Since \( h = 0 \), there are no particular costs for accepting a job long before it will be processed by the system and having it waiting for its turn very long. If \( c < -r(0) \), then the limiting strategy accepts all jobs, i.e., \( p = 1 \). In that case the optimal (termination) strategy will prescribe to serve a part of all jobs up to and including phase \( k \) and the rest up to and including phase \( k + 1 \), where \( k = \lfloor \frac{p}{c} \rfloor \) and \( k + 1 = \lceil \frac{p}{c} \rceil \). (If \( \frac{p}{c} \) takes an integer value, \( k \) say, then all jobs in the system are granted \( k \).)
phases.) If \( c > -r(0) \), then, in general, it is not optimal to accept all jobs, but just a fraction \( p < 1 \). These jobs are served up to and including some phase \( k \), where \( \lambda p^k = 1 \).

**Discounted profit case** In case of discounting we can also identify several cases. If \( c > -r(0) \), then rejecting a job gives a higher direct reward than accepting it. If the number of jobs in the system is large, it will take too long before we can get compensation for this immediate loss. So, again, the problem is essentially finite. If \( c < -r(0) \), then jobs will always be accepted if the number of jobs in the system is very large and the horizon \( n \) is sufficiently far away.

### 7 Counterexamples

If the reward function \( r(k) \) is indeed non-decreasing in \( k \), but not concave, then, in general, Theorem 1 does not hold. By means of the following three counterexamples, it is demonstrated subsequently that part 2, both parts of part 1, as well as part 4 of Theorem 1 need not hold.

**Counterexample 1** Let \( N = 7 \) and let the rewards be defined as

\[
\mathbf{r} = (r(0), \ldots, r(7)) = (0, 4, 4, 10, 10, 15, 15, 19).
\]

Let \( \mu = 1 \) (so \( \alpha = 0 \), \( \lambda = 0 \)) and \( h = 0 \), so there is no discounting and there are no future arrivals nor holding costs.

Let the remaining number of periods be 3. We are interested in the optimal policy in case the state is \((i, 4)\), \( i \geq 1 \), i.e., there is at least one job in the system and the job under service resides in node 4.

Noting that since \( \alpha = 0 \) and \( \lambda = 0 \), service completion events are the only events that can occur, it is not hard to figure out that for \( i = 1, 2, 3 \), the (unique) optimal policies and corresponding (expected) profits \( W_3(i, 4) \) are as follows.

<table>
<thead>
<tr>
<th>( i )</th>
<th>first decision</th>
<th>next decision</th>
<th>final decision</th>
<th>( W_3(i, 4) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>continue</td>
<td>continue</td>
<td>continue</td>
<td>19</td>
</tr>
<tr>
<td>2</td>
<td>abort</td>
<td>continue</td>
<td>continue</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>continue</td>
<td>abort</td>
<td>abort</td>
<td>23</td>
</tr>
</tbody>
</table>

From the second column of the table (displaying the decisions made at \( n = 3 \)) it can be concluded that part 2 of Theorem 1 does not hold.

**Counterexample 2** Consider the same data as in Counterexample 1. Let \( c = 2 \). Then \( W_3(1, 4, \text{arr}) = 19 \), with unique optimal first decision reject. However, \( W_3(2, 4, \text{arr}) = 21 \), with unique optimal first decision accept, violating the first part of part 1 of Theorem 1, and \( W_3(1, 3, \text{arr}) = 18 \), also with unique optimal first decision accept, violating the second part of part 1 of Theorem 1. Therefore, both parts of part 1 of Theorem 1 do not hold.
Counterexample 3  Consider the same data as in Counterexample 1, but with $h = 1$ and $n = 1$. Furthermore, let $c = 0$. Then $W_1(1, 2, \text{arr}) = 9$, with unique optimal decision reject, but $W_1(2, 2) = 8$, with unique optimal decision continue. This violates part 4 of Theorem 1.

8 Conclusions

We have considered a single server workload model with Poisson arrivals and Erlang service times. For this $M|E_N|1$ queue we have dealt with two additional decision features. First, one has to decide upon arrival to accept or reject the job. Second, one can decide to quit serving a job. The reward for a job for which only $k$ phases have been completed is $r(k)$. This reward function is assumed to be non-decreasing and concave. We have shown that the optimal strategy for both types of decisions is characterized by threshold policies, viz, reject if the system is considered to be full and quit serving if there is too much work waiting and if the job under service has already passed a sufficient number of service phases.

The research is motivated by issues arising in workflow control, which captures the on-line decision making process in workflow management. For in-depth discussions of workflow management, we refer to Lawrence [2]. For an elaborate treatment of some quantitative analytical aspects of workflow management, we refer to Brouns [1]. Our workload model covers an important aspect of most workflow problems, namely the fact that there is not enough capacity to treat all jobs completely.

Workflow problems encountered in practice are far more complicated. We intend to study at least the following three extensions: real finite time, other phase type service distributions and multi servers. We expect that a better understanding of the optimal strategies for these problems will help us to formulate good strategies for more general workflow problems.

References
