Queueing Systems (2024) 108:475–577
https://doi.org/10.1007/s11134-024-09923-0

Core allocation to minimize total flow time in a multicore system in the presence of a processing time constraint

Rein Vesilo[1]

Received: 10 May 2022 / Revised: 23 July 2024 / Accepted: 25 July 2024 /
Published online: 26 August 2024

Abstract

Data centers are a vital and fundamental infrastructure component of the cloud. The requirement to execute a large number of demanding jobs places a premium on processing capacity. Parallelizing jobs to run on multiple cores reduces execution time. However, there is a decreasing marginal benefit to using more cores, with the speedup function quantifying the achievable gains. A critical performance metric is flow time. Previous results in the literature derived closed-form expressions for the optimal allocation of cores to minimize total flow time for a power-law speedup function if all jobs are present at time 0. However, this work did not place a constraint on the makespan. For many diverse applications, fast response times are essential, and latency targets are specified to avoid adverse impacts on user experience. This paper is the first to determine the optimal core allocations for a multicore system to minimize total flow time in the presence of a completion deadline (where all jobs have the same deadline). The allocation problem is formulated as a nonlinear optimization program that is solved using the Lagrange multiplier technique. Closed-form expressions are derived for the optimal core allocations, total flow time, and makespan, which can be fitted to a specified deadline by adjusting the value of a single Lagrange multiplier. Compared to the unconstrained problem, the shortest job first property for optimal allocation is maintained; however, a number of other properties require revising and other properties are only retained in a modified form (such as the scale-free and size-dependence properties). It is found that with a completion deadline the optimal solution may contain groups of simultaneous completions. In general, all possible patterns of single- and group-completion need to be considered, producing an exponential search space. However, the paper determines analytically that the optimal completion pattern consists of a sequence of single completions followed by a single group of simultaneous completions at the end, which reduces the search space dimension to being linear. The paper validates the Lagrange multiplier approach by verifying constraint qualifications.

✉ Rein Vesilo
rein.vesilo@mq.edu.au

[1] School of Engineering, Macquarie University, Sydney, NSW 2109, Australia

Keywords Data center · Multicore · Speedup function · Flow time · Optimization

Mathematics Subject Classification 68M20 · 60K30 · 90C30

1 Introduction

In its press release of November 10, 2021, Gartner states that the "Cloud will be the centerpiece of new digital experiences". The importance of the cloud can be measured by the predictions in the press release that in 2022, global cloud revenue will increase to $US474 billion, from $US408 billion in 2021 [1]. The cloud, as a third-party service, can be used as a service, platform, or infrastructure in both offline and time-sensitive online modes [2]. The core element of the cloud is the data center, which provides the physical space to house the thousands of servers and storage and networking equipment needed, together with ancillary services such as cooling. Performance analysis of such systems is of widespread interest and benefit.

The requirement for data centers to execute large numbers of demanding jobs places a premium on resources, including processing capacity. In multicore systems (where a "core" is a generic processing unit), parallel programming techniques enable sequentially programmed jobs to be reconfigured to execute on multiple cores to reduce flow time, defined as the time a job spends in the system; this is a critical performance metric used in data centers.

However, the flow time reductions resulting from parallelization can vary. Much of the variation is due to application-specific dependent factors such as communication and synchronization requirements between threads, input–output demands, checkpointing, the presence of serial execution components in a job, and so on. A common method for representing the dependence between flow time and the number of cores is through the speedup function. The speedup function, in this paper, is defined as the execution speed using a given core allocation relative to when all cores are used. It is an increasing function, the property that captures the increased effectiveness of using more cores, and it is often a sublinear and concave function, the property that represents the reduced marginal benefit of adding additional cores [3]. With a power-law speedup function, the functional form of the speedup function is θ^p, where θ is the proportion of cores allocated to a job and $0 \leq p \leq 1$. For highly parallelizable jobs, p is close to 1, while for highly serialized jobs, p is close to 0. The power-law speedup function is a commonly used speedup function [3–5]. In this paper, it is assumed that all jobs have the same power-law speedup function, with the same parameter value p.

Recently, there has been a burst of interest in determining how speedup affects the optimal allocation of cores in multicore systems with parallelizable jobs. This is illustrated by the comprehensive review of open problems in data centers [3] that devoted an extensive discussion to multicore speedup. The review observes that for systems where job sizes are known, optimizing mean response time is still "largely an open problem". The case of a power-law speedup function is one of the most tractable cases. Specific results for this case are obtained by [6, 7]. These papers derive closed-form expressions for the optimal allocation of cores to minimize weighted total flow

time in a multicore system with all jobs present at time 0 for a power-law speedup function.

Core allocation with parallelizable jobs involves tradeoffs between different scheduling methods. Without completion deadlines [6] shows that core allocation requires a tradeoff between shortest remaining processing time (SRPT) scheduling and processor sharing (PS) scheduling. The SRPT aspect seeks to complete jobs as quickly as possible by prioritizing short jobs, whereas the PS aspect seeks to improve efficiency by processing jobs in the most efficient part of the speedup curve, which is why the allocation algorithm in [6] was called high-efficiency SRPT (heSRPT). By utilizing the shape of the speedup function and exploiting the regularity of the power law, [6] derived the following properties of heSRPT:

- Jobs complete in order of *shortest job first (SJF)*, which preserves the SRPT aspect of allocation.
- Core allocations only change at the time of a job completion.
- Core allocations display the *scale-free* property: the ratios of core allocations of uncompleted jobs remain constant irrespective of the number of jobs in the system.
- Core allocations are *universally quantized*: the possible allocations belong to the same discrete set of values across all possible combinations of job sizes.
- Core allocations display *size independence*: once the job order is known, the job sizes do not affect the allocations, even if a particular job's size is slightly varied, provided the completion order is not altered.

However, the models used by [6, 7] assume that there are no constraints on the completion time. For many applications, such as financial services, social networking, and web search areas, latency targets are essential to ensure that users receive responses fast enough to avoid an adverse impact on their perceived experiences (with late responses being either discarded or delivered incompletely, e.g., less relevant search results) [8–12]. The impact of latency violations can lead to significant reductions in service operator revenue.

As time constraints were not considered in [6, 7], there is the possibility that, using the core allocations derived in those papers, unacceptable completion deadline violations may occur. The aim of this paper is to address this missing but important aspect of multicore speedup analysis to analyze the impact of completion deadlines for the case of a power-law speedup function with all jobs present at time 0, and derive closed-form expressions for the flow time and the makespan. This paper is the first to date, to the author's knowledge, to investigate this case and obtain analytic results.

1.1 Contributions

The paper derives closed-form mathematical expressions for the minimum total flow time and the optimal ordering of jobs in the presence of a completion deadline that is assumed to be the same for all jobs. In the process, the paper expands in several directions the fundamental concepts and methods required to determine core allocations in multicore systems. As a result, the advances made in this paper contribute significantly to the ongoing understanding of an open area of data center performance analysis.

The paper by [6] extensively exploited the scale-free allocation property to derive closed-form expressions. However, in the presence of a completion deadline, it is not clear how the scale-free property can be applied. One would reasonably expect that a scale-free property still applies: but in what form, how does it depend on the completion deadline; and, how can it be used to obtain the optimal allocations?

A key factor is makespan, defined as the time required to complete all jobs. When there is no completion deadline, determining the makespan is almost secondary: it can be computed after all the allocations have been determined. However, in the presence of a completion deadline, knowing the relationship between the makespan and the completion deadline is essential for ensuring that jobs complete within the specified time limit. The challenge is that this is a chicken-and-egg type problem: knowing the correct allocations requires knowing the makespan, but knowing the makespan requires knowing the correct allocations, hence, complicating the scale-free property.

For this reason, in this paper a different optimization approach is used from that in [6]. Instead, a nonlinear optimization program is formulated and solved using the Lagrange multiplier technique. Although this technique is well known, the constraints of the optimization problem are unusual in that the dependence between job size and job processing time is implicitly represented. However, by using this approach and solving the associated Karush–Kuhn–Tucker (KKT) conditions, the exact dependence of allocations on the completion deadline can be expressed using a single Lagrange multiplier variable. This simplification facilitates the derivation of optimal core allocations. The paper finds that the solution to the optimization problem may be one in which the jobs are completed individually or in groups. Group completion is akin to the minimal makespan solution, but not necessarily involving all jobs. Group completion does not occur in the optimal solution when the completion deadline is absent. The *completion pattern* specifies the sequence of job completions and how they are grouped. For each possible completion pattern, expressions for the total flow time and makespan are derived as finite sums of simple terms with job sizes as coefficients. The use of the Lagrange multiplier technique and KKT conditions is rigorously validated through an analysis of constraint qualifications. It is found that these are satisfied for $0 < p < 1$. The case $p = 1$ corresponds to SRPT, which is widely covered in the existing literature (see, for example, [13, 14]).

Using these derivations, we show that some properties of heSRPT remain unchanged, other properties are modified, and other properties no longer apply. In particular,

- The allocation decision still involves a tradeoff between SRPT and PS. However, a third tradeoff is required involving the makespan that combines the possibility of single and multiple simultaneous job completions (see Sect. 2.4).
- Jobs still complete in SJF order.
- Core allocations only change at job completion epochs.
- The scale-free property is retained, but in a modified form.
- Core allocations are no longer universally quantized. Across all possible job size combinations, almost any core allocation is realizable.
- Size independence is preserved, but in a modified form.

The paper then analytically determines which completion pattern gives the smallest total flow time. A source of complexity of the presence of groups is that the search space for completion patterns has an exponential dimension in the number of jobs. The paper, however, is able to negotiate this complexity by analytically deriving ordering relations between completion patterns and uses these to prove that optimal completion pattern has a simple form. In particular, the optimal completion pattern is shown to be one comprising a sequence of individual job completions followed by a single group of completions at the end. The optimal completion pattern is the one with the longest sequence of individual completions that is viable, with longer patterns not being viable. Therefore, the search complexity is linear in the number of jobs. A further increase in search efficiency is obtained by exploiting the ordering that exists between completion patterns to enable the search to be terminated once the boundary between viable and unviable completion patterns in the linear search has been located.

The ordering results used in this paper to derive the optimal completion pattern are obtained by using what is called a *split transformation* in which a group of simultaneous completions is divided into two groups. It is shown that this leads to a decrease in total flow time. An appropriately chosen sequence of split transformations then enables two completion patterns to be compared, thereby allowing the optimal completion pattern to be determined. In general, a split transformation can increase or decrease the makespan. The paper shows that makespan always increases for fixed-size jobs. Such jobs may occur in practice and provide a limiting case of the analysis. Further, from a theoretical perspective, the paper proves that fixed-size jobs minimize the decrease in total flow time that can be achieved under broad conditions, elaborated later.

The final contribution of the paper relates to the $p = 1$ case. In this case, the processing rate equals 1 and the system is work conserving in the sense defined by [15]. If jobs complete individually, SRPT is the optimal scheduling algorithm [3, 7]. Since all jobs are present at time 0, SRPT is equivalent to SJF. However, with the parallel execution of jobs using continuously divisible cores, it is possible for jobs to complete simultaneously. We extend the standard SRPT results to derive the minimum total flow time for any completion pattern. SJF is still the optimal ordering of job completions. We verify that SRPT is the solution with the smallest total flow time over all completion patterns (see Sect. 6.7).

The analysis of systems with arrivals and speedup functions is still an open problem and is a very challenging area for multicore speedup analysis [3]. All results to date have relied on simplifying assumptions for the speedup function or scheduling algorithm (see Related work, Sect. 1.2). With online arrivals, when a new job arrives, previous jobs may still be in the system with work to complete. Part of the optimization problem is determining the allocations needed for continuing and new jobs where the time of arrival of the next job provides a completion deadline. One of the benefits of the results developed in this paper is that they will lead to potentially new techniques for determining these allocations when power-law speedup functions are used.

The results of this paper can also be extended to wireless networking systems. The paper [16] obtains the same allocation equations as heSRPT (in the unweighted case) for a wireless communication service system. In this context, jobs correspond to data flows, the speedup function corresponds to rate reductions due to interference between

channels, and the allocation constraints arise from the capacity region. The results in this paper can be used to extend [16] to obtain optimal capacity allocations when a flow completion deadline is imposed.

The paper is organized as follows. Section 1.2 presents related work. The key concepts used in the paper are described in Sect. 2. The optimization problem for minimizing total flow time and its formulation are presented in Sect. 3. A high-level road map for solving the problem and the statement of the main theorem (Theorem 4.1) are given in Sect. 4. The optimization problem is divided into two subproblems. The first subproblem determines the optimal order of jobs (described in Sect. 5), while the second subproblem determines the optimal allocation of cores for the optimal order of jobs. Solving the second subproblem requires solving the KKT conditions and the determination of the optimal completion pattern. This is described in Sects. 6 through 9 (see the road map for more high-level details.) The conclusion is given in Sect. 10. Most of the detailed derivations are contained in the Appendices.

1.2 Related work

Parallelization is widely used at scales up to and beyond massively parallel (including embarrassingly parallel) (see [17] for a review in the high-performance computing context). Typical applications that utilize parallelization include: The Map-Reduce framework [18], which is widely used to significantly increase productivity in data-intensive online systems in cluster systems and multicore environments [19], and the Google TensorFlow [20] software library platform that is used to support parallel processing in machine learning (including deep learning and convolutional neural networks), image processing, and synthetic radar aperture imagery applications [21]. Another form of multicore system involves graphics processing units (GPUs). For example, [22] investigate the performance of a deep learning algorithm using a GPU with 1152 cores.

The benefits of using speedup functions to analyze multicore systems was demonstrated by [23]. Speedup functions for different multicore architectures are discussed by [5] from the perspective of Amdahl's law. With Amdahl's law [24] the speedup, S, obtained when using a resource that gives a speedup of s for fraction p of the time is given by $S = \frac{1}{(1-p)+\frac{p}{s}}$. To apply this in a multicore system, one core is used to execute the sequential part, while the remaining cores are used to execute the parallelizable part.

The review by [3] on data center computing includes a review of parallelization in multicore systems and speedup functions. Tradeoffs in core allocation include efficiency drops if too many servers are allocated to a single job. Also, bias toward small job sizes can reduce the mean response time. The review also discusses the suitability of the power-law speedup function for modeling real performance data, using PARSEC-3 parallel benchmarks [25]. Besides the power-law speedup function, a threshold speedup function is discussed where the speedup function is $s(k) = k$ $(k \leq k_0, s(k) = k_0\ (k > k_0)$ to model systems where the gains of parallelization cannot be extended beyond a given number of cores. The review observes that different types of job may require different speedup functions.

Latency targets are usually specified in service level agreements (SLAs) and can be specified as a strict deadline to be enforced or using a probability target for not exceeding a deadline. Latency is an important metric in IoT (Internet-of-Things) cloud computing applications for intelligent management of devices in real-time with back-end processing through virtual machines in the infrastructure as a service model. [26] incorporate deadlines into a management framework to meet SLAs that include time limits, soft deadlines, and average missed deadlines. They present a range of schedulers to meet deadlines. [27] examine efficient scheduling of jobs with time-dependent utility functions while achieving fairness, and [28] devise a scheduler called earliest maximal waiting time first (EMWTF) to meet job deadlines. However, multicore speedup dependence is not considered in such schedulers.

Closed-form expressions for the optimal allocation of cores to minimize average slowdown, where the slowdown of a job is defined as the ratio of the actual job flow time to the job flow time when all cores are used, were determined by [7]. The same paper generalizes these results to determine optimal core allocations to minimize the weighted total flow time when the weights decrease with job size.

The review by [3] examines the allocation of cores with the arrival of jobs. Policies described in the paper include the EQUI policy and the GREEDY policy. The EQUI policy divides processing cores equally between jobs so as to operate in the most efficient part of the speedup curve and minimizes the mean response time for Poisson arrivals and exponential job sizes. However, it is not optimal when job sizes are known. In contrast, the GREEDY policy allocates cores to achieve the maximum departure rate of jobs at all times from the system. [29] analyzes systems with arrivals with elastic and inelastic jobs with the goal of minimizing the mean system time. The paper shows that if the inelastic jobs have a smaller mean size than the elastic jobs, then the inelastic jobs should have priority. [30] propose a number of heuristic algorithms that can be used to schedule jobs arriving at a server system by running a stochastic gradient descent optimization application. One algorithm is called High-Efficiency Low Latency (HELL) which equalizes job efficiency, where a job's efficiency is defined as the fraction $s(k)/k$, where $s(k)$ is the speedup function. Another algorithm in [30] is called KNEE, which determines the number of cores a job requires before the marginal reduction in efficiency falls below a threshold.

2 Key concepts

2.1 Jobs

The paper considers a multicore processing system that can process multiple jobs in parallel. The capacity to parallelize jobs can exist in different forms. With so-called moldable jobs, variable numbers of cores can be allocated prior to when execution begins, but once a job is running, the number of cores remains unchanged. With malleable jobs, not only can the number of cores be selected prior to execution, but the number can be adjusted during execution [31–33]. With perfect malleability, it is assumed that there is no time penalty for adjusting the number of cores during execution. This is the assumption made in this paper.

All jobs are assumed to be present at time 0, with J jobs in total, and the jobs are numbered $1, 2, \cdots, J$. The size of the job j, denoted by x_j, is defined as the time it would take for the job to complete if all the cores of the system were available for the job. The sizes of the jobs are assumed to be known in advance. To avoid trivialities, it is assumed that all jobs are of nonzero size. It is assumed that there is no upper bound on the number of jobs in the system that can be processed at any given time. The completion deadline, t_0, specifies the time by which all the jobs must have been completed. The same deadline applies to all jobs. Given a sequence of jobs $x_k, \cdots, x_n$, define $\widetilde{x}_{k:n} \equiv \left(\sum_{i=k}^{n} {x_i}^{1/p}\right)^p$. For a vector $\boldsymbol{y}$, the r-norm of the vector is given by $\|\boldsymbol{y}\|_r = \left(\sum_i y_i^r\right)^{1/r}$. Therefore, $\widetilde{x}_{k:n} = \|(x_k, \cdots, x_n)\|_{1/p}$.

2.2 Total flow time

Once all the work associated with a job has been completed, the cores it used are released; the job is then free to leave the system. The completion time of job j is denoted by T_j. The flow time (system time) of job j is the amount of time the job has spent in the system. Since all jobs are present at time 0, this equals T_j. The total flow time over all jobs is given by

$$S = \sum_{j=1}^{J} T_j.$$

The average flow time of a job is $\sum_{j=1}^{J} T_j/J$ and minimizing average flow time is equivalent to minimizing total flow time. The makespan is given by

$$T = \max_j T_j.$$

2.3 Speedup function

A system with an integer number, N, of processing cores is considered. In real systems, cores, which we consider to be the basic units of processing capacity, and assumed to be of equal capacity in this paper, are allocated to jobs in integer quantities. Suppose k cores are allocated to a job. The normalized core allocation for the job is defined by $\theta = k/N \in [0, 1]$. For mathematical analysis, the requirement that cores are indivisible resource units is relaxed and we assume that cores can operate at continuous fractional capacities. With this assumption θ is a continuous-valued quantity[1]. A similar assumption is used by [7]. The rate at which work can be done using k cores is defined by the unnormalized speedup function, $S(k)$. Homogeneity is assumed so that the same function, $S(k)$ applies to all jobs. The rate at which work is performed using k cores relative to when all cores are used is given by $S(k)/S(N)$. The normalized *speedup function* (or just speedup function) is defined as $s(\theta) \equiv S(\theta N)/S(N)$. For

[1] Alternatively, for large systems ($N \to \infty$), we can use the asymptotic limit of the real system normalized allocation k/N.

small allocations, the assumption of continuously divisible cores can become an issue, and the discrete nature of resources should be taken into account. For example, [3] suggests, for $k < 1$, an unnormalized speedup function given by $S(k) = k$. In this case, real processor architectures become important (see, for example, [24]).

A job of size x given a fixed allocation θ of cores will take time

$$t_{\text{process}}(x) = \frac{x}{S(\theta N)} = \frac{x}{s(\theta)S(N)} \tag{1}$$

to process. For a fixed number N, of cores and given speedup function, the normalized processing time will be $x/s(\theta) = t_{\text{process}}(x)S(N)$. In general, if a job is allocated θ cores for the time interval $[t_1, t_2]$, then the normalized work[2] performed on the job in the interval is $(t_2 - t_1)s(\theta)$, provided that the job does not complete in the interval. The normalizing constant $S(N)$ in (1) becomes significant when comparing different speedup functions. For example, suppose S_1 and S_2 are two speedup functions for which the normalized processing time is smaller for S_1, but if $S_2(N) > S_1(N)$ then it is possible for S_2 to have the smaller actual processing time.

Clearly, to minimize total flow time, there will be no unallocated cores if there are jobs in the system, and the system is said to be fully utilized in this sense. It follows that if there are m jobs in the system labeled $1, \cdots, m$, and the core allocations for these m jobs are $\theta_1, \cdots, \theta_m$, then full utilization implies $\theta_1 + \cdots + \theta_m = 1$.

This paper assumes that the speedup function is given by a power law in which the processing rate for a job using k cores is given by a function of the form $S(\theta N) = (\theta N)^p$, for a given $0 \leq p \leq 1$ with all jobs having the same value of p. The speedup function is thus $s(\theta) = \theta^p$. If p is close to 1, then the jobs are highly parallelizable, with the maximum speedup occurring for $p = 1$; for $p = 1$ the speedup function is linear. If p is close to 0 then the jobs are highly serialized. In this case, $s(\theta)$ is almost a constant close to 1 for most θ allocations,[3] and only small increases in speedup can be achieved by using more cores. For $p = 0$, no speedup occurs and a job can be processed using a single core (in the continuous core allocation case, a job is allocated an arbitrarily small amount of cores).

2.4 Illustrative example

This section explains the intuition behind the makespan and total flow time tradeoff. With no time constraints, the optimal allocation decision reduces to heSRPT. However, in the presence of a completion deadline, allocation using heSPRT may produce a completion time violation. This violation can be avoided by reducing the makespan. This is illustrated in the following simple example.

For simplicity, suppose that $p = 0.5$ and that there are $J = 2$ jobs, labeled job 1 and job 2, of equal size 1, in the system. The initial allocations for jobs 1 and 2 are $\theta_{1,1}$ and $\theta_{2,1}$, such that $\theta_{1,1} + \theta_{2,1} = 1$, and these apply until job 1 is completed. Job 2

[2] Henceforth, normalized work will be referred to as just work.

[3] More specifically, (in our idealized setting), $s(\theta)$ is said to be close to 1 if $s(\theta) > 1 - \epsilon$ for $\theta > \delta$), for some small value of ϵ (dependent on how close we want $s(\theta)$ to be to 1).

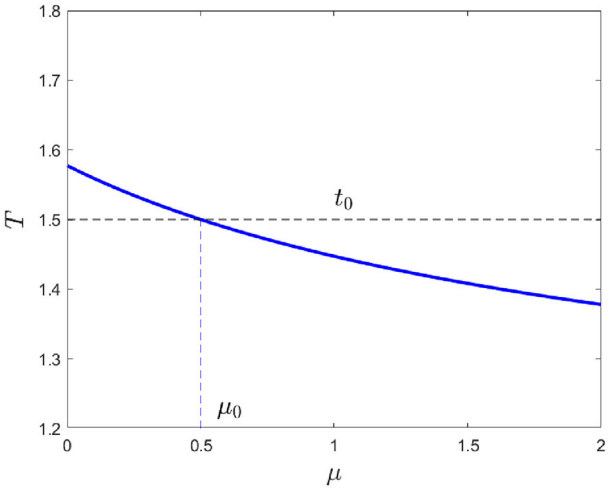

Fig. 1 Two jobs: makespan

is then allocated all the cores to complete its processing with the allocation $\theta_{2,2} = 1$[4]. The optimal allocation using heSRPT without a completion deadline is $\theta_{1,1} = 0.75$, $\theta_{2,1} = 0.25$ and $\theta_{2,2} = 1$. (An unusual aspect of heSRPT is that for jobs of equal size, the optimal allocation can be strongly unequal.) The makespan can be determined as $1+1/\sqrt{3} = 1.5774$. If the completion deadline $t_0 < 1+1/\sqrt{3}$ then the solution of the unconstrained problem is not viable. On the other hand, simultaneous completion of the two jobs has a makespan of $\sqrt{2} = 1.412$. What the current paper shows is that for $\sqrt{2} \le t_0 < 1+1/\sqrt{3}$ it will still be possible to obtain a viable solution. It will be shown that this can be achieved by introducing a Lagrange multiplier, μ, and expressing the makespan, $T(\mu)$, as a function of μ so that the solution of $T(\mu) = t_0$, denoted by μ_0, enables the t_0-constraint to be met. This is shown in Fig. 1 for $t_0 = 1.5$ giving $\mu_0 = 0.5$. For this value of μ_0 the optimal allocations are $\theta_{1,1} = 0.64$, $\theta_{2,1} = 0.36$ and $\theta_{2,2} = 1$ (see Fig. 2) This allocation reduces the makespan by giving more cores to the job that had the smaller allocation $\theta_{2,1}$ so that allocations are closer to the simultaneous completion case, where $\theta_{1,1} = \theta_{2,1} = 0.5$. However, this can only be done at the expense of an increase in the total flow time, S (see Fig. 3). However, μ cannot be increased indefinitely: it cannot be increased past the point where $\theta_{1,1}$ and $\theta_{2,1}$ cross in Fig. 2, because this will cause the time to complete the processing of job 2 to be negative.[5] Generalizing to more than two jobs, simultaneous completion of all jobs will produce the smallest makespan, but this is at the expense of an increase in total flow time. The results of the paper show that by having only a subset of jobs completed simultaneously, the increase in total flow time can be made smaller. The paper formulates and solves an optimization problem to achieve the above.

[4] See Sect. 3 for a full description of the system model.

[5] This is described in more detail later.

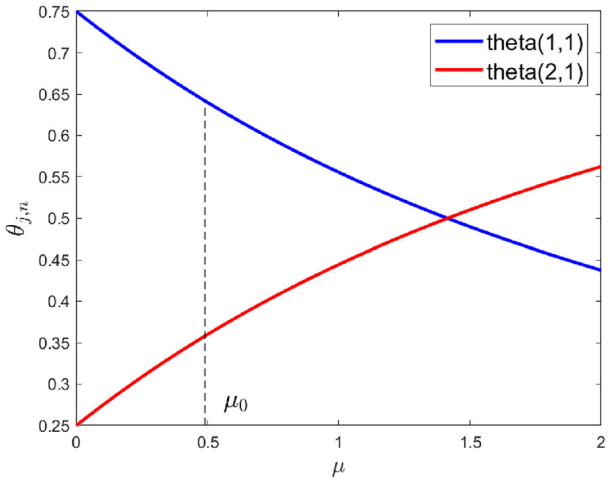

Fig. 2 Two jobs: optimal core allocations

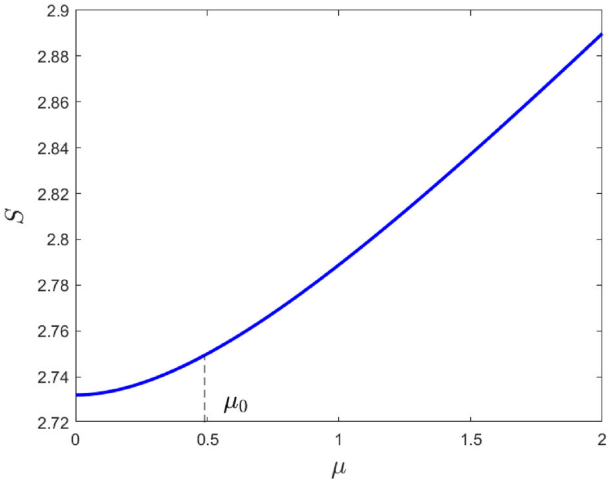

Fig. 3 Two jobs: total flow time

3 Problem formulation

The role of the core allocation component of a multicore system is to determine the number of cores given to each job at each point in time. The criterion used in this paper for core allocation is to minimize the total flow time without violating the completion deadline.

Although job processing proceeds in continuous time, it is sufficient to determine core allocations only at job completion times. This follows along similar lines to Theorem 1 and Appendix A of [7] which proved that the optimal allocations remain

constant between completion times when the criteria is to minimize the sum weighted flow time (in the current paper, all weights are set to 1).[6]

Jobs are completed at successive points in time, but not necessarily in the order that they are indexed. Define T_n as the time of the n'th job completion. The interval $(T_{n-1}, T_n]$ is called *stage* n. The duration of stage n is given by $\tau_n = T_n - T_{n-1}$, where $T_0 = 0$. With J jobs, the number of stages is J. Define the notation $\boldsymbol{\tau} = (\tau_1, \cdots, \tau_J)$.

Suppose job j completes at the end of stage $\pi(j) = n$. The core allocation given to job j in stage n is defined as $\theta_{j,n}$. Define $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_J)$, where $\boldsymbol{\theta}_n = (\theta_{1,n}, \cdots, \theta_{J,n})$.

The approach used in this paper for obtaining the optimal core allocations is to formulate a nonlinear constrained optimization problem. The optimization problem for minimizing total flow time can be formulated as follows:

$$\min_{\{\boldsymbol{\tau},\boldsymbol{\theta},\pi\}} \quad S = \sum_{j=1}^{J} T_{\pi(j)} \tag{2}$$

$$\text{subject to} \quad \sum_{j=1}^{J} \theta_{j,n} = 1, \qquad n = 1, \cdots, J, \tag{3}$$

$$\sum_{n=1}^{J} s_{j,n}\tau_n = x_j, \qquad j = 1, \cdots, J, \tag{4}$$

$$\sum_{n=1}^{J} \tau_n \le t_0, \tag{5}$$

$$\theta_{j,n} \ge 0, \qquad n = 1, \cdots, J,\ j = n, \cdots, J, \tag{6}$$

$$\tau_n \ge 0, \qquad n = 1, \cdots, J. \tag{7}$$

The constraint (3) ensures that all processing cores are used at every stage. The constraint (4) ensures that all work requirements for a job are met and that no more work is done than necessary on a job; this constraint implicitly relates stage durations to core allocations. The constraint (5) ensures that all jobs are completed on or before the specified completion deadline t_0. The constraint (6) ensures that all allocations are nonnegative. The constraint (7) ensures that the duration of the stages is not negative, which implies that negative work cannot be done in (4).

In the above formulation, no assumptions have been made about the completion order—the n'th completion does not necessarily correspond to the job n. In fact, all completion orders are viable since, for any given completion order, all cores can be allocated to the next job scheduled for completion. It is also possible that jobs may complete simultaneously with other jobs; this will occur if $\tau_n = 0$ for one or more

[6] The proof of this is by assuming that if the allocations are non-constant during the interval $[t_1, t_2]$ for which there are no completions then this produces a contradiction. It can be shown, using the concavity of the speedup function, that allocating each job its average allocation in $[t_1, t_2]$ (which is a feasible allocation) strictly reduces the flow time of at least one job without increasing the flow time of any other jobs, thus reducing the sum weighted flow time. This argument still applies in the presence of the completion deadline t_0.

values of n. Observe that $S = \sum_{j=1}^{J} T_{\pi(j)} = \sum_{j=1}^{J} T_j$ (where j is the job index in the first sum, and j is the completion index in the second sum).

4 Road map

This section provides a high-level description of the solution approach. In general, the problem in (2) lacks sufficient structure to obtain a closed-form solution: There are $J!$ job orders that need to be considered, leading to a potential combinatorial explosion of complexity, which is further increased by the various combinations of active and inactive constraints. To manage this complexity, the problem in (2) is divided into two subproblems.

4.1 First subproblem

The first subproblem determines the optimal order of completion of the jobs. This is shown to be shortest job first order and is obtained in Sect. 5.

4.2 Second subproblem

The second subproblem determines the optimal allocations for the optimal job completion order. The statement and solution of the second subproblem are given in 6. It begins by presenting the problem formulation for the SJF completion order in Sect. 6.1. For the case $0 < p < 1$, the second subproblem is solved using the Lagrange multiplier technique. The Lagrangian is given by (19) in Sect. 6.2, and the KKT conditions are presented in Sect. 6.3.

Deriving the solutions of the KKT conditions in the presence of inequality constraints requires consideration of which constraints are active and which constraints are inactive. If $\tau_n = 0$, for some values of n, there will be one or more groups of simultaneous completions (this is a consequence of the makespan versus flow time tradeoff that occurs in the solution of the optimization problem when a completion deadline is imposed.) Recalling that $\tau_1 > 0$, there are $J-1$ τ_n-constraints ($n = 2, \cdots, J$) each of which can be active or inactive. Thus, there are a total of 2^{J-1} arrangements of active and inactive τ_n-constraints. A *completion pattern* refers to a particular sequence of active and inactive τ_n-constraints which describe how completions are grouped. Solving the second subproblem requires determining the optimal completion pattern and the core allocations for that optimal completion pattern. This is divided into two steps:

1. Obtaining the solution to the KKT conditions for a given completion pattern.
2. Determining the optimal completion pattern by comparing the total flow times for different completion patterns.

4.2.1 Solution of the KKT conditions

The solution of the KKT conditions for a given completion pattern is presented in 6.4. The Lagrangian contains four multipliers: λ_n for the core allocation constraints (3),

σ_j for the job size constraints (4), μ for the completion deadline constraints (5), and ν_n for the stage duration constraints (7). The constraints (6) are relaxed. The solution can be expressed in terms of the system primitives and μ. The solution gives

- The optimal core allocation $\theta_{j,n}(\mu)$ for job j in stage n ($n \leq j$) (Proposition 6.1).
- The duration of stage n, $\tau_n(\mu)$, for the optimal core allocations (Proposition 6.2).
- The makespan, $T(\mu)$, for the optimal core allocations (Proposition 6.3).
- The total flow time, $S(\mu)$, for the optimal core allocations (Proposition 6.4).

4.2.2 Obtaining a viable solution

The expressions obtained by solving the KKT conditions for a given completion pattern are expressed as functions of the single argument μ and these functions are well defined for $\mu \in [0, \infty)$. They have the same form regardless of whether the constraint conditions are satisfied or not. For this reason, this solution obtained by formally solving the KKT conditions is referred to as a *potential solution*.

For a potential solution to become a *viable solution*, the constraint conditions must be satisfied. This is described in Sect. 6.5. For each constraint, there is a set of values of μ for which it is satisfied. The intersection of these sets of values is the set of values of μ that give a viable solution. We show that only the t_0 deadline constraint and the $\tau_n \geq 0$ constraints must be explicitly considered, since the other constraints are incorporated into the solution or can be proven to be satisfied. This is examined in Sect. 6.5. For simplicity, it is assumed that $\psi_n = 1$.

If the t_0 deadline constraint is satisfied for some $\mu \in [0, \infty)$, we say that the solution is t_0-viable. t_0-viability is determined by solving the equation

$$T(\mu) = t_0. \tag{8}$$

If the solution of this equation satisfies $\mu \geq 0$ then the t_0 constraint is active. In this case, the quantity μ_0 is defined as the value of μ that satisfies (8). If $T(0) < t_0$, then the t_0-constraint is inactive, and we define μ_0 as 0. In either case, the deadline constraint is satisfied and we say that the solution is t_0-viable. If $t_0 < T(\mu)$ for $\mu \in [0, \infty)$ then (8) cannot be satisfied and the solution is not t_0-viable. (8) has no solution. Proposition 6.5 presents the conditions for t_0-viability used later in the paper. The optimal total flow time for a t_0-viable solution is given by $S(\mu_0)$.

If all the $\tau_n > 0$ constraints are satisfied, we say that the solution is τ_n-viable. Proposition 6.6 shows that, for the SJF job order, the only stage duration constraint that can ever be violated is that of the final completion epoch (with the index denoted by M).

Although viability requires both t_0-viability and τ_n-viability, our proofs at times may involve solutions that are not τ_n-viable since the functions must be considered for values of μ outside the ranges of viability. For example, it is possible to satisfy $T(\mu_0) = t_0$, and yet have $\tau_M(\mu_0) < 0$.

4.2.3 Validity of solution and special cases

The paper proves the validity of the Lagrange multiplier approach by performing an analysis of constraint qualifications, and these are shown to be satisfied. This is discussed in Sect. 6.6.

For $p = 1$ and $p = 0$, the optimal solution can be obtained without using Lagrange multipliers. The solutions for these two cases are given in Sects. 6.7 and 6.8, respectively.

4.2.4 Optimal completion pattern

Sections 7–9 determine the optimal completion pattern. Section 7 presents the notation and concepts used to describe patterns. Given M ($1 \leq M \leq J$), define *scenario* M as the completion pattern where $\tau_n > 0$ ($n \leq M$) and $\tau_n = 0$ ($n > M$). That is, the pattern consists of $M - 1$ single completions followed by a single group of $J - M + 1$ completions at the end (which may be a single completion if $J = M$). We show that if $t_0 < \widetilde{x}_{1:J}$ then no pattern is t_0 viable (Theorem 4.1(i)), and if $\widetilde{x}_{1:J} \leq t_0$ then all patterns are t_0-viable (Theorem 4.1(ii)). If $\widetilde{x}_{1:J} \leq t_0$, then define scenario M^* as the largest value of M such that scenario M is τ_n-viable. The main theorem the paper proves is the following.

Theorem 4.1 *(i) If $t_0 < \widetilde{x}_{1:J}$, then no pattern is t_0-viable.*
(ii) If $\widetilde{x}_{1:J} \leq t_0$, then all patterns are t_0-viable.
(iii) For case (ii), the total flow time for all patterns is minimized by scenario M^*.

The proofs of (i) and (ii) are given by Proposition 9.1. The proof of (iii) is obtained using what is called a *split transformation*. A split transformation is one that takes a completion pattern where there is an index k, where multiple simultaneous completions occur (> 1) and creates a new pattern by dividing this group into two groups of completions of sizes k_1 and k_2 such that $k = k_1 + k_2$. If the next group of completions (which can include a single completion) occurs at index n (or set $n = J + 1$ if no such group exists), then one group of completions, of size k_1, will be at the original index k and the second group of completions, of size k_2, will be at an index l ($k < l < n$). In an end-split transformation, the final group of completions is divided, otherwise the transformation is called an internal-split transformation. The starting equations to determine the change in makespan and total flow time for an internal-split transformation are given at the end of Sect. 7. The equations for an end-split transformation can be obtained as a special case of those for an internal-split transformation.

The proof of Theorem 4.1(iii) is obtained first for the restricted case where the only patterns considered are scenarios and then for all patterns. The proof for scenarios is given in Sect. 8. This is achieved by using end-split transformations to create the entire sequence of scenarios. The end-split transformation introduces a total ordering between scenarios in the sequence that allows the optimal scenario to be determined.

The optimal completion pattern is then determined for all patterns. This is done in Sect. 9. Although an internal-split transformation can be used to convert one pattern into another, it only produces a partial ordering of patterns. To determine the optimal pattern, the approach used is to consider a given pattern and apply internal-split transformations to obtain a sequence of patterns that terminate at a scenario and then use the total ordering of scenarios to compare the total flow time of the starting pattern with the terminating scenario.

The proofs for scenarios and all patterns proceed through similar steps. In each case, one or more sequences of patterns are considered. For each sequence, intervals on the makespan axis are derived that allow the t_0- and τ_n-viability of the patterns in the sequence to be determined. These intervals are determined before the t_0-constraint is applied. When a particular value of t_0 is introduced it is positioned relative to the intervals to determine the t_0- and τ_n-viability of each pattern in the sequence. The final step is to determine the change in total flow time across the entire pattern sequence.

The paper finishes by examining the case of minimizing total flow time in which there is no completion deadline and how the results described in the paper can be used to solve the problem of minimizing makespan.

In the sections that follow, precise statements of the above road map are given. Detailed aspects of the proofs are deferred to the appendices. The technical lemmas used in the proofs are given at the end of the appendices.

5 First subproblem: optimal job completion order

In general, solving the first subproblem requires the consideration of $J!$ possible job completion orders. For the unconstrained problem, Theorem 2 of [7] proved that the optimal completion order in a problem to minimize the sum weighted flow time was SJF (for the current paper, all weights are equal.) When a completion deadline is present, it is now shown that the optimal completion order is also SJF.

Starting with the proof given in [7], which is by contradiction, the proof for the constrained case supposes that there are two jobs S (denoting "small") and L (denoting "large") with respective job sizes, x_S and x_L, such that $x_S < x_L$ with respective completion time T_S and T_L but that $T_L < T_S$. The proof in [7] shows that by transferring an appropriate amount of cores from the L job to the S job the S job can be made to complete at time T_L and the L job can be made to complete at time $T_L^* < T_S$ without adversely affecting any other jobs, thus reducing the sum weighted flow time. In the presence of the completion deadline, t_0, the same interchange can be used as the completion times will either remain unchanged or be reduced. It is possible, however, that the interchange may change the completion order of jobs. This will cause a relabeling of the indices in the formulation of the problem in (2) with the $\tau_n \geq 0$ constraints maintained.

6 Second subproblem: optimal core allocations for SJF completion order

6.1 Problem formulation for SJF completion order

Having determined in the first subproblem that the optimal job completion order is SJF, the second subproblem computes the optimal core allocations for SJF ordering of jobs. To avoid cumbersome notation, jobs are relabeled so that the job index corresponds to the job completion order, so that, after relabeling, $\pi(j) = j$. With this relabeling, job n completes at time T_n and the duration of stage n is $\tau_n = T_n - T_{n-1}$ ($T_0 = 0$). Since a job cannot be completed at time 0, it will always be the case that $\tau_1 > 0$. If jobs j and $j+1$ complete simultaneously, then $T_n = T_{n+1}$ and $\tau_{n+1} = 0$. Because jobs are indexed in completion order,

$$T_j = \sum_{n=1}^{j} \tau_n, \tag{9}$$

the makespan is

$$T = T_J = \sum_{n=1}^{J} \tau_n \tag{10}$$

and the total flow time is

$$S = \sum_{j=1}^{J} T_j = \sum_{j=1}^{J} \sum_{n=1}^{j} \tau_n = \sum_{n=1}^{J} (J - n + 1)\tau_n. \tag{11}$$

The core allocation given to job j during stage n is $\theta_{j,n}$ and the processing speed of job j during stage n is $s_{j,n} \equiv s(\theta_{j,n})$. In stage n, only jobs $n, \cdots, J$ require core allocations since jobs $1, \cdots, n-1$ have completed; implying $\theta_{j,n} = 0$ for $j < n$. The work remaining for job j at the end of stage n is defined as $r_{j,n}$, where $r_{j,0} = x_j$, by definition. The one-step update equation for $r_{j,n}$ is

$$r_{j,n} = (r_{j,n-1} - \tau_n s_{j,n})^+, \tag{12}$$

where $x^+ = \max(0, x)$. If $r_{j,n} = 0$, then the job n has been completed at the end of stage n, otherwise $r_{j,n} > 0$. After relabeling, $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_J)$ where $\boldsymbol{\theta}_n = (\theta_{n,n}, \cdots, \theta_{J,n})$, the optimization problem for minimizing the total flow time in (2) becomes:

$$\min_{\{\boldsymbol{\tau}, \boldsymbol{\theta}\}} \sum_{n=1}^{J} (J - n + 1)\tau_n \tag{13}$$

$$\text{subject to} \quad \sum_{j=n}^{J} \theta_{j,n} = \psi_n, \qquad n = 1, \cdots, J, \tag{14}$$

$$\sum_{n=1}^{j} s_{j,n}\tau_n = x_j, \qquad j = 1, \cdots, J, \tag{15}$$

$$\sum_{n=1}^{J} \tau_n \leq t_0, \tag{16}$$

$$\theta_{j,n} \geq 0, \qquad n = 1, \cdots, J, \, j = n, \cdots, J, \tag{17}$$

$$\tau_n \geq 0, \qquad n = 1, \cdots, J. \tag{18}$$

The problem formulation has been generalized by introducing the constraint on number of cores in stage n in (14) to be ψ_n, rather than 1.[7] The constraint equations (14) and (17) ensure that $\theta_{j,n} \leq \psi_n$. The formulation of the above problem corresponds to that of [6], with the exception of the additional t_0-constraint.

The problem in (13) is a geometric programming optimization problem and can be converted into a convex optimization problem by suitable function and variable transformations (see [34] p. 162). However, this approach is not adopted in this paper.

6.2 Solution of the second subproblem

The paper [6] determined the optimal allocations using algebraic methods that exploited the scale-free property. However, attempting to apply the scale-free property in the presence of a completion deadline is more difficult because it is not obvious how the scale-free property depends on the presence of a completion deadline. Further, because the possibility of groups of completions has been introduced, it is not clear how this affects the scale-free property. For these reasons, the paper adopts the Lagrange multiplier method to solve the optimization problem. Using the Lagrangian, KKT conditions are derived that are then solved to obtain the optimal core allocation, makespan, and total flow time. It is shown that there is a Lagrange multiplier (μ) such that once this is known, the allocation expressions have a structure similar to those determined in [6].

To reduce the number of Lagrange multipliers, the problem is relaxed by not including the $0 \leq \theta_{j,n}$ constraints in (17). It will be shown that for the optimal allocations determined, the constraints in (17) are satisfied. The Lagrangian function for the optimization problem in (13)–(18) is then given by

[7] It could be claimed that constraint (14) should an inequality constraint; however, with the exception of $p = 0$, it is clear that the constraint should be active to maximize the work done.

$$\mathscr{L}(\boldsymbol{\tau}, \boldsymbol{\theta}, \Lambda) = \sum_{n=1}^{J}(J-n+1)\tau_n + \sum_{n=1}^{J}\lambda_n\left(\sum_{j=n}^{J}\theta_{j,n} - \psi_n\right) + \sum_{j=1}^{J}\sigma_j\left(x_j - \sum_{n=1}^{j}s_{j,n}\tau_n\right) + \mu\left(\sum_{n=1}^{J}\tau_n - t_0\right) + \sum_{n=1}^{J}\nu_n(-\tau_n)\,,$$

where λ_n are the Lagrange multipliers for the constraint in (14), σ_j are the Lagrange multipliers for the job size constraints in (15), μ is the Lagrange multiplier for the completion deadline in (16), ν_n are the Lagrange multipliers for the stage duration constraints in (18), and $\Lambda \equiv (\lambda_1, \cdots, \lambda_J, \sigma_1, \cdots, \sigma_J, \mu, \nu_1, \cdots, \mu_J)$. To facilitate analysis, the Lagrangian is rearranged as follows:

$$\mathscr{L}(\boldsymbol{\tau}, \boldsymbol{\theta}, \Lambda) = \sum_{n=1}^{J}(J-n+1)\tau_n + \sum_{n=1}^{J}\sum_{j=n}^{J}\lambda_n\theta_{j,n} - \sum_{n=1}^{J}\lambda_n\psi_n + \sum_{j=1}^{J}\sigma_j x_j - \sum_{n=1}^{J}\tau_n\sum_{j=n}^{J}\sigma_j s_{j,n} + \mu\left(\sum_{n=1}^{J}\tau_n - t_0\right) + \sum_{n=1}^{J}\nu_n(-\tau_n)\,. \tag{19}$$

6.3 KKT conditions

Obtaining the KKT conditions proceeds by taking partial derivatives $\mathscr{L}$ (in (19)) with respect to the primal variables. Taking partial derivatives of $\mathscr{L}$ with respect to $\theta_{j,n}$ and equating to 0 gives, for $n = 1, \cdots, J$, $j = n, \cdots, J$,

$$\frac{\partial}{\partial\theta_{j,n}}\mathscr{L}(\boldsymbol{\tau}, \boldsymbol{\theta}, \Lambda) = \lambda_n - \sigma_j\tau_n\frac{\partial s_{j,n}}{\partial\theta_{j,n}} = 0\,. \tag{20}$$

Taking partial derivatives of $\mathscr{L}$ with respect to τ_n and equating to 0 gives, for $n = 1, \cdots, J$,

$$\frac{\partial}{\partial\tau_n}\mathscr{L}(\boldsymbol{\tau}, \boldsymbol{\theta}, \Lambda) = (J-n+1) + \mu - \nu_n - \sum_{j=n}^{J}\sigma_j s_{j,n} = 0. \tag{21}$$

The complementary slackness conditions are

$$\mu \geq 0, \tag{22}$$

$$\nu_n \geq 0, \tag{23}$$

Table 1 Completions epochs for $J = 6$, pattern = 100110

Completion epoch	Completion index	Jobs completing
1	1	1,2,3
2	4	4
3	5	5,6

$$\mu \left(\sum_{n=1}^{J} \tau_n - t_0 \right) = 0, \tag{24}$$

$$\nu_n \tau_n = 0. \tag{25}$$

6.4 Solution of KKT conditions for a given completion pattern

This section presents the solution to the KKT conditions for a given completion pattern. A detailed derivation of these results is given in Appendix A.

6.4.1 Completion patterns

Recall that a completion pattern refers to a particular sequence of active and inactive τ_n-constraints and describes how completions are grouped. This is now described in detail. First observe that if $\tau_n > 0$ then the τ_n-constraint is inactive and $\nu_n = 0$ under the KKT conditions, while if $\tau_n = 0$ then the τ_n-constraint is active, and $\nu_n \geq 0$.

A completion pattern can be represented by an ordered tuple $b = (b_1, b_2, \cdots, b_J)$ where $b_j = 0$ if $\tau_j = 0$ and $b_j = 1$ if $\tau_j > 0$. Since $\tau_1 > 0$, $b_0 = 1$ always.

A *completion epoch* is defined as a point in time where a group of one or more simultaneous completions occur. The time point of a completion epoch coincides with the end of a stage of positive duration. This stage is followed by zero or more stages of zero duration, whereby each completion in the group corresponds to one of these stages, including the one of positive duration. For example, if $\tau_m > 0$ and $\tau_{m+1}, \cdots, \tau_{m+k-1} = 0$, then jobs $m, \cdots, m + k - 1$ are completed simultaneously at the time of the end of stage m. The index of a completion epoch is defined as the smallest stage index in the group of completions. In the example just given, the index of the completion epoch is m. The index of the d'th completion epoch is denoted by $\hat{d}$ and is the d'th value of n, in sequence, such that $\tau_n > 0$, so that $\hat{d} = n$. Define D as the total number of completion epochs and define M as the index of the final completion epoch, so that $\hat{D} = M$. An example showing which jobs are completed at each completion epoch for $J = 6$ is shown in Table 1 for the completion pattern 100110. In this example, $D = 3$ and $M = 5$, and we have $\widehat{1} = 1, \widehat{2} = 4$, and $\widehat{3} = 5$.

We now introduce the notation for describing completions at adjacent epochs and groups of completions. Given a stage index n, define $\underline{n}$ as the largest index less than or equal to n such that $\tau_n > 0$. This means that if jobs $m, \cdots, m + k - 1$ complete simultaneously ($\tau_m > 0$, $\tau_{m+1} = \cdots = \tau_{m+k-1} = 0$) and $m \leq n \leq m + k - 1$, then $\underline{n} = m$ is the index of the first job in the group. In particular, if $\tau_n > 0$, then $\underline{n} = n$.

Define $\overline{n}$ as the index of the next completion epoch. In the above example, if $\tau_{m+k} > 0$, then $\overline{n} = m + k$. If $n = M$, then no such index exists, and we define $\overline{M} = J + 1$. Finally, define $\underline{\underline{n}}$ as the largest index less than $\underline{n}$ such that $\tau_n > 0$. If $n = 1$, then no such index exists and we define $\underline{\underline{n}} = 0$. Single and multiple job completions can now be characterized:

- Single job completion. This happens if $\tau_n > 0$, and $n < J$ and $\tau_{n+1} > 0$ or $n = J$. In this case, the completing job number j is n. Then we have $\underline{n} = n$ and $\overline{n} = n+1$.
- Multiple job completions. Suppose that job j is one of a group of jobs that are completed simultaneously. The group of jobs that are completed simultaneously with the job j is $\underline{j}, \cdots, \overline{j} - 1$, so that $\underline{j} \le j \le \overline{j} - 1$. In this case, $\tau_{\underline{j}} > 0$ and $\tau_{\underline{j}+1} = \cdots = \tau_{\overline{j}-1} = 0$.

Recall that $\widetilde{x}_{k:n} \equiv \left(\sum_{i=k}^{n} {x_i}^{1/p}\right)^p$ and, for a vector $\boldsymbol{y}$, the r-norm of the vector is given by $\|\boldsymbol{y}\|_r = \left(\sum_i y_i^r\right)^{1/r}$. Therefore, $\widetilde{x}_{k:n} = \|(x_k, \cdots, x_n)\|_{1/p}$. In particular, if jobs $\underline{n}, \cdots, \overline{n}$ are completed simultaneously, $\widetilde{x}_{\underline{n}:\overline{n}-1} = \left(\sum_{j'=\underline{n}}^{\overline{n}-1} {x_{j'}}^{1/p}\right)^p = \|(x_{\underline{n}}, \cdots, x_{\overline{n}-1})\|_{1/p}$. For a single completion, $\underline{n} = n$ and $\overline{n} = n + 1$ to give $\widetilde{x}_{\underline{n}:\overline{n}-1} = x_n$. For notational convenience, define also: $x_0 = 0$.

6.4.2 Optimal core allocations

Assume that $0 < p < 1$ and define $c = 1/(1 - p)$. In the following, we assume that μ has a given value that will be determined later. Define

$$\alpha_n \equiv \frac{(J - n + 1) + \mu}{\psi_n^p}. \tag{26}$$

For notational convenience, define $\alpha_{J+1} = 0$.

If $\tau_n > 0$, then Proposition 6.1 solves the KKT conditions to give the optimal core allocation for j in stage n ($n \le j$).

Proposition 6.1 *If $\tau_n > 0$, then the optimal core allocation for job j in stage n ($n \le j$) is given by*

$$\theta_{j,n} = \frac{\sigma_j^c}{\alpha_n^c}\psi_n, \tag{27}$$

where if a single job completes at the end of stage n, then

$$\sigma_n = \begin{cases} (\alpha_n^c - \alpha_{n+1}^c)^{1/c} & n < J \\ \alpha_J & n = J \end{cases}; \tag{28}$$

and if jobs $\underline{n} \le n \le \overline{n}$ complete simultaneously then

$$\sigma_n^c = (\alpha_{\underline{n}}^c - \alpha_{\overline{n}}^c)^p \left(\frac{x_n}{\widetilde{x}_{\underline{n}:\overline{n}-1}}\right)^{1/p}. \tag{29}$$

Proof See Appendix A.1. □

Observe that (29) reduces to (28), when applied to a single completion. The quantities α_n and σ_j play a central role in the calculation of allocations and performance quantities. They satisfy the following identities (see Lemma A.3)

$$\sum_{j=n}^{J} \sigma_j^c = \alpha_n^c \qquad n = 1, \cdots, J. \tag{30}$$

Analogously to the relationship between $\widetilde{x}_{1:n}$ and $\{x_1, \cdots, x_n\}$ we define $\widetilde{\sigma}_{l:n-1} \equiv (\sigma_l^c + \cdots + \sigma_{n-1}^c)^{1/c}$. (Observe that $\widetilde{\sigma}_{M:J} = \alpha_M$.) From (30), $\widetilde{\sigma}_{l:n-1} = (\alpha_l^c - \alpha_n^c)^{1/c}$, allowing (29) to be expressed as

$$\sigma_j^c = \widetilde{\sigma}^c_{\underline{j}:\overline{j}-1} \left(\frac{x_j}{\widetilde{x}_{\underline{j}:\overline{j}-1}} \right)^{1/p}.$$

6.4.3 Scale-free property

Eq. (27) implies that if two jobs j and j' are both being processed in stage n, then

$$\frac{\theta_{j,n}}{\theta_{j',n}} = \frac{\sigma_j^c}{\sigma_{j'}^c}. \tag{31}$$

This is independent of the stage index n, and is what defines the *scale-free property* of the system: Even as the allocations change from stage to stage, the ratios of the allocations between jobs j and j' for different stages are unchanged. This property was first observed in [6]. Lemma H.6(i) shows, for $j < j'$, that $\sigma_j > \sigma_{j'}$. This means that jobs that are completed earlier have a greater impact on total flow time than those that are completed later.

6.4.4 Size dependence of allocations

For a single completion, Eqs. (27) and (28) give

$$\theta_{j,n} = \frac{\alpha_j^c - \alpha_{j+1}^c}{\alpha_n^c} \psi_n. \tag{32}$$

For a fixed value of μ, the optimal allocation is independent of the size of the job. By contrast, for multiple simultaneously completing jobs, (27) and (29) give the following.

$$\theta_{j,n} = \frac{\alpha_{\underline{j}}^c - \alpha_{\overline{j}}^c}{\alpha_n^c} \left(\frac{x_{j'}}{\widetilde{x}_{\underline{j}:\overline{j}-1}} \right)^{1/p} \psi_n. \tag{33}$$

In this case, for a given value of μ, the optimal allocation depends on the size of the job. However, the sum of allocations for all simultaneously completing jobs is size independent, equal to $(\alpha^c_{\underline{j}} - \alpha^c_j)/\alpha^c_n$. This preserves in a modified form the size-independence property of unconstrained optimization. However, for a given completion deadline, the value of μ, and hence α_n, will depend on t_0, which in turn depends on the sizes of jobs (see (36), below), and therefore the optimal allocations will ultimately depend on job sizes. This is in contrast to the unconstrained case, where optimal allocations are always size-independent for a given job completion order.

6.4.5 Stage durations

For the optimal allocation, the duration of stage n ($\tau_n > 0$) is given by

Proposition 6.2 *For $\tau_n > 0$,*

$$\tau_n = \left(\frac{x_n}{\sigma_n^{cp}} - \frac{x_{\underline{\underline{n}}}}{\sigma_{\underline{\underline{n}}}^{cp}} \right) \frac{\alpha_n^{cp}}{\psi_n^p} = \left(\frac{\widetilde{x}_{n:\overline{n}-1}}{(\alpha_n^c - \alpha_{\overline{n}}^c)^p} - \frac{\widetilde{x}_{\underline{\underline{n}}:n-1}}{(\alpha_{\underline{\underline{n}}}^c - \alpha_n^c)^p} \right) \frac{\alpha_n^{cp}}{\psi_n^p}. \tag{34}$$

Proof See Appendix A.2. □

This result can be expressed using completion epoch numbers. Suppose that the index of the d'th completion epoch occurs is n, so that $\widehat{d} = n$. Noting that $\underline{\underline{n}} = \widehat{d-1}$, and $\overline{n} = \widehat{d+1}$, this gives

$$\tau_{\widehat{d}} = \left(\frac{\widetilde{x}_{\widehat{d}:\widehat{d+1}-1}}{(\alpha^c_{\widehat{d}} - \alpha^c_{\widehat{d+1}})^p} - \frac{\widetilde{x}_{\widehat{d-1}:\widehat{d}-1}}{(\alpha^c_{\widehat{d-1}} - \alpha^c_{\widehat{d}})^p} \right) \frac{\alpha^{cp}_{\widehat{d}}}{\psi^p_{\widehat{d}}}.$$

6.4.6 Makespan

The makespan for the optimal allocation is given below[8].

Proposition 6.3

$$T = \sum_{n=1,\tau_n>0}^{J} \tau_n \tag{35}$$

$$= \sum_{n=1,\tau_n>0}^{J} \left(\frac{\widetilde{x}_{n:\overline{n}-1}}{(\alpha_n^c - \alpha_{\overline{n}}^c)^p} - \frac{\widetilde{x}_{\underline{\underline{n}}:\widehat{d}-1}}{(\alpha_{\underline{\underline{n}}}^c - \alpha_n^c)^p} \right) \frac{\alpha_n^{cp}}{\psi_n^p}. \tag{36}$$

[8] There is a slight abuse of notation in (35) (and similar later expressions) in that the condition $\tau_n > 0$ in the sum conditions indicates that index n is the first job in a group of simultaneous job completions. In a viable solution, this will be the case. However, for unviable solutions, if index M is included in the sum then it is possible that $\tau_M < 0$. (See Lemma 6.3.)

This can be rearranged to give

$$T = \widetilde{x}_{M:J} + \sum_{n=1,\tau_n>0}^{M-1} \frac{\widetilde{x}_{n:\overline{n}-1}}{\left(\alpha_n^c - \alpha_{\overline{n}}^c\right)^p} \left(\frac{\alpha_n^{cp}}{\psi_n^p} - \frac{\alpha_{\overline{n}}^{cp}}{\psi_{\overline{n}}^p} \right). \tag{37}$$

Proof See Appendix A.3. □

Similarly to stage duration, makespan can be expressed in terms of completion epoch numbers. For example, (37) can be expressed as

$$T = \sum_{d=1}^{D} \tau_{\widehat{d}} = \widetilde{x}_{\widehat{D}:J} + \sum_{d=1}^{D-1} \frac{\widetilde{x}_{\widehat{d:d+1}-1}}{\left(\alpha_{\widehat{d}}^c - \alpha_{\widehat{d+1}}^c\right)^p} \left(\frac{\alpha_{\widehat{d}}^{cp}}{\psi_{\widehat{d}}^p} - \frac{\alpha_{\widehat{d+1}}^{cp}}{\psi_{\widehat{d+1}}^p} \right).$$

6.4.7 Total flow time

Expressions for the total flow time are obtained similarly to those for makespan.

Proposition 6.4

$$\begin{aligned} S &= \sum_{n=1,\tau_n>0}^{J} (J-n+1)\tau_n \qquad (38) \\ &= (J-M+1)\widetilde{x}_{M:J} \\ &\quad + \sum_{n=1,\tau_n>0}^{M-1} \frac{\widetilde{x}_{n:\overline{n}-1}}{\left(\alpha_n^c - \alpha_{\overline{n}}^c\right)^p} \left((J-n+1)\frac{\alpha_n^{cp}}{\psi_n^p} - (J-\overline{n}+1)\frac{\alpha_{\overline{n}}^{cp}}{\psi_{\overline{n}}^p} \right). \qquad (39) \end{aligned}$$

Proof See Appendix A.4. □

Remark 1 We observe that the method of the obtaining the solution to (13) can be extended to obtain the formal solution to the more general problem of minimizing the objective function $\sum_{n=1}^{J} w_n \tau_n$, where w_n $(n = 1, \cdots, J)$ is a sequence of weights, where the problem has the same job completion order and the same constraints as in (13). In this case, the KKT conditions (21) becomes $w_n + \mu - \nu_n \sum_{j=n}^{J} \sigma_j s_{j,n} = 0$ $(n = 1, \cdots, J)$. The solution to this new problem can be derived following the same steps as in Appendix A but using $\alpha_n = w_n + \mu$. It will produce the same expressions for τ_n in terms of α_n as in (34), from which the expressions for T and S can be obtained directly. However, the viability of the solution still needs to be determined.

6.5 Viability of a potential solution

For a given completion pattern, the form of the equations for τ_n, T and S in a potential solution does not depend on whether the t_0-constraint and $\tau_n \geq 0$ constraints are satisfied or not. This section presents conditions to determine the viability of a potential solution.

6.5.1 Equations for $\psi_n = 1$

For the remainder of the paper, the simplifying assumption of $\psi_n = 1$ is made. For this case, τ_n, T and S become the following.

The duration of stage n, for the case $\tau_n > 0$, is given by, after setting $\psi_n = 1$ in (34),

$$\tau_n(\mu) = \left(\frac{\widetilde{x}_{n:\overline{n}-1}}{\sigma^{cp}_{n:\overline{n}-1}} - \frac{\widetilde{x}_{\underline{\underline{n}}:n-1}}{\sigma^{cp}_{\underline{\underline{n}}:n-1}} \right) \alpha^{cp}_n, \tag{40}$$

where $\sigma^c_{n:\overline{n}-1} = \alpha^c_n - \alpha^c_{\overline{n}}$ and $\sigma^c_{\underline{\underline{n}}:n-1} = \alpha^c_{\underline{\underline{n}}} - \alpha^c_n$.

For later use, two special cases for computing τ_n are for $n = 1$ and $n = M$. These values are recorded in the following lemma.

Lemma 6.1 *(i)*

$$\tau_1 = \begin{cases} \frac{\widetilde{x}_{1:\overline{1}-1}}{(\alpha^c_1 - \alpha^c_{\overline{1}})^p} \alpha^{cp}_1, & M > 1 \\ \widetilde{x}_{1:J}, & M = 1 \end{cases}, \tag{41}$$

where, for $M > 1$, $\overline{1}$ $(= \widehat{2})$ is the index of the second completion epoch.

(ii)

$$\tau_M = \begin{cases} \widetilde{x}_{M:J} - \widetilde{x}_{\underline{\underline{M}}:M-1} \frac{\alpha^{cp}_M}{(\alpha^c_{\underline{\underline{M}}} - \alpha^c_M)^p}, & M > 1 \\ \widetilde{x}_{1:J}, & M = 1 \end{cases}, \tag{42}$$

where, for $M > 1$, $\underline{\underline{M}}$ $(= \widehat{D-1})$ is the index of the second last completion epoch.

Proof (i) For $n = 1$, $\underline{\underline{n}} = 0$, $\overline{n} = \overline{1}$ and $x_{\underline{\underline{n}}:n-1} = x_0 = 0$. Inserting this into (40) gives the case for $M > 1$. If $M = 1$ then $\overline{n} = J + 1$ for which $\alpha_{J+1} = 0$, to give $\tau_1 = \widetilde{x}_{1:J}$.

(ii) Here, $n = M$, and $\overline{n} = J + 1$. This gives $\sigma^c_{n:\overline{n}-1} = \alpha^c_M - \alpha^c_{J+1} = \alpha^c_M$, giving the case for $M > 1$. If $M = 1$, then $\underline{\underline{M}} = 0$ and $x_{\underline{\underline{M}}:M-1} = x_0 = 0$, to give $\tau_M = \widetilde{x}_{1:J}$ (i.e., $\tau_1 = \tau_M$).

□

The equation for makespan, (37), becomes

$$T(\mu) = \widetilde{x}_{M:J} + \sum_{n=1, \tau_n > 0}^{M-1} \widetilde{x}_{n:\overline{n}-1} \frac{\alpha^{cp}_n - \alpha^{cp}_{\overline{n}}}{\widetilde{\sigma}^{cp}_{n:\overline{n}-1}}. \tag{43}$$

Lemma A.8 shows that the coefficient of each term $\widetilde{x}_{n:\overline{n}-1}$ $(n \leq M-1)$ in T is strictly increasing. Therefore, for $M > 1$, $T(\mu)$ is a strictly decreasing function of μ for $M > 1$, and thus $T(0) \geq T(\mu) > T(\infty)$. The maximum value of T $(\mu \geq 0)$ is

$$T(0) = \widetilde{x}_{M:J} + \sum_{n=1,\tau_n>0}^{M-1} \widetilde{x}_{n:\overline{n}-1} \frac{(J-n+1)^{cp} - (J-\overline{n}+1)^{cp}}{((J-n+1)^c - (J-\overline{n}+1)^c)^p}. \tag{44}$$

For $\mu \to \infty$, Lemma H.4(i)(c) shows the coefficient of each term $\widetilde{x}_{n:\overline{n}-1}$ ($n \le M-1$) in T has limit 0. Therefore, $\lim_{\mu\to\infty} T(\mu) = \widetilde{x}_{M:J}$. For $M = 1$, $T(\mu)$ is a constant function with value $\widetilde{x}_{1:J}$.

The equation for total flow time (39) becomes

$$S(\mu) = (J-M+1)\widetilde{x}_{M:J} + \sum_{n=1,\tau_n>0}^{M-1} \widetilde{x}_{n:\overline{n}-1} \frac{(J-n+1)\alpha_n^{cp} - (J-\overline{n}+1)\alpha_{\overline{n}}^{cp}}{\widetilde{\sigma}_{n:\overline{n}-1}^{cp}}. \tag{45}$$

Lemma A.9 shows that the coefficient of each term $\widetilde{x}_{n:\overline{n}-1}$ ($n \le M-1$) in S is strictly increasing, and so, for $M > 1$, $S(\mu)$ is an strictly increasing function of μ. For $M = 1$, $S(\mu)$ is a constant function with value $J\widetilde{x}_{1:J}$.

6.5.2 Conditions for viability

For a given completion pattern, T, S, and τ_n are expressed as functions of the single argument μ, which is the Lagrange multiplier associated with the t_0-constraint. The functions are well defined for $\mu \in [0, \infty)$. For the solution of the KKT conditions to become a *viable solution* a suitable value of μ must be determined which satisfies all the constraints in the optimization problem (13) and the complementary slackness conditions (22)–(25). Each constraint defines a set of values of μ for which it is satisfied. The intersection of these sets of values is the set of values of μ that gives a viable solution.

t_0-constraint

The following lemma shows that if the t_0 constraint is satisfied, then it is satisfied by a single value of μ that is denoted by μ_0. The only exception is if $M = 1$.

Lemma 6.2 *(i) Suppose $M > 1$. The value of μ that satisfies the t_0-constraint is given by μ_0:*

$$\mu_0 = \begin{cases} 0, & T(0) \le t_0, \\ \textit{The solution of } T(\mu) = t_0, & T(\infty) < t_0 \le T(0), \\ \textit{undefined}, & t_0 \le T(\infty). \end{cases} \tag{46}$$

(ii) For $M = 1$,

$$\mu_0 = \begin{cases} 0, & T(0) < t_0, \\ 0, & t_0 = T(\infty), \\ \textit{undefined}, & t_0 < T(\infty). \end{cases} \tag{47}$$

Proof (i) From the complementary slackness conditions (22) and (24), $\mu > 0$ or $\mu = 0$, depending on whether the constraint is active or inactive, respectively. For $M > 1$, Lemma A.8 shows that $T(\mu)$ is a strictly decreasing function of μ on $[0, \infty)$, so that $T(\infty) < T(\mu) \leq T(0)$. Note that the limit value $T^M(\infty)$ cannot be attained for $0 \leq \mu < \infty$. This gives the following cases:

(a) If $T(0) < t_0$ then the t_0-constraint is inactive. In this case, $\mu_0 = 0$.
(b) If $T(0) = t_0$, then the constraint t_0 is weakly active.[9] In this case, $\mu_0 = 0$.
(c) If $T(\infty) < t_0 \leq T(0)$ then the t_0-constraint is active. In this case, the value of $\mu_0 \geq 0$ is obtained by solving $T(\mu_0) = t_0$. The value of μ_0 can be obtained by means of a numerical solution.
(d) If $t_0 \leq T(\infty)$ then there is no solution to $T(\mu_0) = t_0$ and thus there is no t_0-viable solution.

(ii) If $M = 1$ then $T(\mu) = \widetilde{x}_{1:J}$. If $t_0 = T(\infty)$, the solution of $T(\mu) = t_0$ can be achieved by any $\mu \geq 0$, in which case we arbitrarily set $\mu_0 = 0$. The other cases of $T(0) < t_0$ and $t_0 < T(\infty)$ are straightforward.

□

If the remaining constraint conditions are satisfied, then the makespan of the optimal solution is given by $T(\mu_0)$ and the total flow time is given by $S(\mu_0)$, where μ_0 is given by (46) ($M > 1$) and (47) ($M = 1$). The t_0-viability of a pattern can be determined by inspecting the value $T(\infty)$.

Proposition 6.5 *A pattern is t_0-viable if and only if either $T(\infty) < t_0$ ($M > 1$) or $T(\infty) \leq t_0$ ($M = 1$).*

Proof This follows immediately from Lemma 6.2. For ease of notation, we will state that the condition for t_0-viability is just $T(\infty) < t_0$ with the understanding that if $M = 1$ then the condition is $T(\infty) \leq t_0$. □

τ_n-constraints

Lemma 6.3, following, shows that the only τ_n-constraint that can be violated is τ_M.

Lemma 6.3 *(i) For any job order, $\tau_1 > 0$.*
(ii) For SJF job order, $\tau_n > 0$ for $1 < n < M$.
(iii) For $M > 1$, the only τ_n-constraint that can be violated is for $n = M$. The τ_n-constraints are satisfied for $0 \leq \mu < \mu_\triangle$, where $\mu_\triangle$ is defined as the solution of $\tau_M(\mu) = 0$. The value of $\mu_\triangle$ is given by

$$\mu_\triangle = \frac{M - \underline{\underline{M}}}{\left(\frac{\widetilde{x}_{\underline{\underline{M}}:J}}{\widetilde{x}_{M:J}}\right)^{(1-p)/p} - 1} - (J - M + 1), \tag{48}$$

in which $\underline{\underline{M}}$ is the index of the second-last completion epoch.
(iv) For $M = 1$, $\tau_M > 0$ and $\mu_\triangle = \infty$.

[9] The weakly active case of $T(0) = t_0$ will be treated as active.

Proof (i) From (41) in Lemma 6.1(i), τ_1 is given by

$$\tau_1 = \frac{\widetilde{x}_{1:j-1}}{(1-(\alpha_j/\alpha_1)^c)^p},$$

where $j = \overline{1}$ is the index of the second completion epoch. Clearly, $\tau_1 > 0$, which is consistent with the assumption that $\tau_1 > 0$ in the formulation of the optimization problem (13). Lemma A.10 proves that τ_1 is an increasing function of μ with $\tau_1 \to \infty$ ($\mu \to \infty$).

(ii) From (40), τ_n ($1 < n < M$) is the difference of two increasing functions. In general, τ_n need not be monotonic; it can initially decrease and then increase. However, Lemma A.11 shows that $\tau_n > 0$ for SJF ordering.

(iii) For $M > 1$, τ_M is given by (42). Lemma A.12(ii)(a) shows that τ_M is a decreasing function of μ with $\tau_M \to -\infty$ ($\mu \to \infty$), which implies that τ_M will be negative if μ is large enough and the constraint will be violated. However, Lemma A.12(ii)(c) shows that $\mu_\triangle > 0$. Therefore, since τ_M is a continuous function, the equation $\tau_M(\mu) = 0$ will have the solution $\mu_\triangle$, and its value is given by (48) (see also Lemma A.12(ii)(d)). Furthermore, for $\mu \in [0, \mu_\triangle)$ $\tau_M(\mu) > 0$ and $\mu \in (0, \infty)$ $\tau_M(\mu) < 0$.
Thus, τ_M is the only stage duration that can become negative, and a pattern is τ_n-viable if and only if $\tau_M(\mu) > 0$. This happens if $\mu < \mu_M$. The end point $\mu = \mu_\triangle$ has been excluded since this will result in the last completion epoch being deleted and the pattern will be different.

(iv) This follows from $\tau_M = \tau_1 > 0$. In this case, the equation $\mu_M(\mu) = 0$ has no solution and we set $\mu_\triangle = \infty$.

□

If a pattern is t_0-viable, then Proposition 6.6 provides a more usable method to determine the τ_n-viability of the pattern. Define $T_\triangle \equiv T(\mu_\triangle)$.

Proposition 6.6 *Suppose $M > 1$. If a pattern is t_0-viable and if jobs have SJF order then the pattern is τ_n-viable, that is, it is (completely) viable, if and only if either $T_\triangle < t_0$ ($M > 1$) or $T_\triangle \le t_0$ ($M = 1$).*[10]

Proof If the t_0-constraint is satisfied, with μ_0 given by Lemma 6.2, then the τ_n-constraint is satisfied if $0 \le \mu_0 < \mu_\triangle$. This is equivalent to the condition $T_\triangle < t_0$, since T is a decreasing function. If the t_0-constraint is not satisfied, then μ_0 is not defined, although $\mu_\triangle$ still exists. In this case, there is no viable solution. □

6.5.3 Other constraints

The remaining two complementary slackness conditions (22) and (25) are satisfied, since it is assumed that $\tau_n > 0$, which implies $\nu_n = 0$. The constraints $\theta_{j,n} \ge 0$ are not problematic, since the optimal allocations given by (27) and (33) for the relaxed

[10] For ease of notation we will state the condition as $T_\triangle < t_0$, with the understanding that if $M = 1$, we can have equality.

problem are all positive. This follows since, for $\mu \geq 0$, and using (26), $\alpha_n > 0$. Also, α_n decreases with n and so by (29), $\sigma_n > 0$. Thus, from (27), $\theta_{j,n} \geq 0$. Since the optimal solution for the relaxed problem meets the constraints for the original unrelaxed problem, it is also the optimal solution for that problem. Following this analysis of the other constraints, the only two constraints that must be checked for viability are the t_0-constraint and the τ_n-constraints.

6.6 Necessary conditions for minimum

The use of KKT conditions is a common technique for obtaining the necessary conditions to determine the minimum point of an optimization problem. However, ensuring that there are no possible minimum points that are not satisfied by the KKT conditions requires additional constraint qualifications. It is shown in Appendix E that the Linear Independence Constraint Qualification (LICQ) is satisfied for $0 < p < 1$ at all points in the feasible set. LICQ also ensures that Lagrange multipliers are unique ([35] p.339).

6.7 $p = 1$

The case $p = 1$ corresponds to the maximum parallelizability of jobs. In this case, the speedup function is linear: $s_{j,n} = \theta_{j,n}$. For $p = 1$, the LICQ conditions are not satisfied. Instead, the solution can be obtained directly from the optimization problem (13).

From Sect. 5, SJF ordering is optimal. With J jobs, there are 2^{J-1} possible completion patterns. The makespan and total flow time for the optimal solution for each possible completion pattern are given by the following proposition.

Proposition 6.7 *(i) Regardless of the completion pattern, the makespan is given by*

$$T = \sum_{j=1}^{J} x_j. \tag{49}$$

(ii) If all jobs are completed individually, then the minimum flow time is given by

$$\sum_{j=1}^{J} (J - j + 1)x_j. \tag{50}$$

The optimal core allocations are $\theta_{n,n} = 1$ $(n = 1, \cdots, J)$ and $\theta_{j,n} = 0$, otherwise. That is, the optimal allocations are SRPT allocations.

(iii) Suppose completions occur individually except for a single completion epoch where multiple jobs complete simultaneously. Suppose this epoch occurs at the end of stage, m, where k jobs are completed simultaneously. For this case, the

minimum flow time is given by

$$S = \sum_{j=1}^{m-1}(J-j+1)x_j + \sum_{j=m}^{m+k-1}(J-m+1)x_j + \sum_{j=m+k}^{J}(J-j+1)x_j. \quad (51)$$

The optimal core allocations are $\theta_{n,n} = 1$ *(*$n < m$, $m+k \le n$*),*

$$\theta_{j,m} = \frac{x_j}{x_m + \cdots + x_{m+k-1}}, \qquad m \le j \le m+k-1,$$

and $\theta_{j,n} = 0$*, otherwise.*

(iv) Suppose there are $I > 0$ *simultaneous completion epochs, such that at the end stage* m_i *there are* k_i *completions (*$i = 1, \cdots, I$*). The minimum total flow time is given by*

$$S = \sum_{j=1}^{J}(J-j+1)x_j + \sum_{i=1}^{I}\sum_{j=m_i}^{m_i+k_i-1}(j-m_i)x_j. \quad (52)$$

Proof See Appendix F. □

Since (49) is independent of the completion pattern, if one completion pattern is t_0-viable, they all are.

From Proposition 6.7(iii), the k simultaneously completing jobs at epoch m are replaced by a single equivalent job of size $x_m + \cdots + x_{m+k-1}$ that has weight m when computing the total flow time; that is, the contribution to the total flow time for these k jobs is

$$\sum_{j=m}^{m+k-1}(J-m+1)x_j. \quad (53)$$

The optimal allocation is a modified all-or-nothing allocation of cores to jobs: if the next job to complete is a single job, it receives all core allocations, while for simultaneously completing jobs, the completing jobs receive all cores, and these are distributed in proportion to the job sizes.

We obtain the following corollary to Proposition 6.7.

Corollary 6.1 *The smallest total flow time, over all completion patterns, occurs when all completions occur individually, with the minimum value given by* (50)*, which is SRPT allocation.*

Proof Over all completion patterns, the smallest minimum total flow time occurs when the right-hand side of (52) is a minimum. This happens when the second term is zero, so that $k_i = 1$, and completions occur individually; total flow time in (52) then equals (50). □

6.8 $p = 0$

For the case $p = 0$, we have $s_{j,n} = 1$, for which there are no parallelizability gains. The solution for $p = 0$ can also be obtained directly from the optimization problem (13). The makespan and total flow time are independent of core allocation and given, respectively, by $T = x_J$ and $S = \sum_{j=1}^{J} x_j$. See Appendix G. Under the assumption of continuously divisible cores, the allocation to any job can be an arbitrarily small positive amount. In practice, the discrete nature of cores must be considered.

7 Optimal completion pattern

This section determines the optimal completion pattern. Recall that scenario M ($1 \leq M \leq J$) refers to a pattern where there are $M - 1$ single completions followed by a group of $J - M + 1$ simultaneous completions. In scenario M, M is the index of the final completion epoch. It is proved (Theorem 4.1(iii)) that for any completion pattern N, either scenario M^* has a shorter total flow time compared to the completion pattern N or that the completion pattern N is not τ_n-viable (and therefore not viable). The intuitive interpretation of the optimality result is that individual completions should be pushed as early as possible, helping to minimize total flow time, and that simultaneous completions should be deferred to the end, helping to reduce the makespan to meet the t_0-constraint.

With 2^{J-1} ways of selecting $\tau_n \geq 0$ ($n = 2, \cdots, J$), there is an exponentially increasing search space of completion patterns. Theorem 4.1 reduces the search space from being exponential to being linear with the number of jobs. A further reduction in complexity occurs due to the ordering between scenarios, because the smallest total flow time occurs for the largest value of M such that scenario M is viable. Starting from $M = 1$ once an unviable scenario has been found, those scenarios with larger values of M can be ignored. Alternatively, the search can begin with $M = J$ and proceed to smaller values of M until the first viable scenario is found, at which point the search can stop.

7.1 Completion pattern transformations

Determining the optimal completion pattern is achieved through what is called a *split transformation*, where a group of completions at an epoch is divided into two groups of completions at successive epochs: one group is at the original completion epoch and the second group is inserted before the subsequent completion epoch in the original completion pattern. Suppose that pattern 1 is the pattern before the split transformation and pattern 2 is the pattern after the split transformation. The superscripts 1 and 2 will be used to distinguish variables belonging to patterns 1 and 2. The following notation is introduced ($i = 1, 2$): $T_0^i = T^i(\mu_0^i)$, $S_0^i = S^1(\mu_0^i)$, $T_\triangle^i = T^i(\mu_\triangle^i)$, and $S_\triangle^i = S^i(\mu_\triangle^i)$.

Using this notation, $\mu_0^1 \geq 0$ and $\mu_0^2 \geq 0$ are the values of μ_0 of the Lagrange multiplier that satisfy the optimization problem (13) for patterns 1 and 2, respectively.

Table 2 End-split transformation ($* =$ can be 0 or 1)

	1	2	3	4	5	6	7
Index		k			l	J	$n = J + 1$
Before	$*$	1	0	0	0	0	
After	$*$	1	0	0	1	0	

Table 3 Internal-split transformation

	1	2	3	4	5	6	7	$\cdots$	J
Index		k			l		n		
Before	$*$	1	0	0	0	0	1	$*$	$*$
After	$*$	1	0	0	1	0	1	$*$	$*$

If the t_0-constraint for pattern $i = 1, 2$ is active, then $T^i(\mu_0^i) = T_0^i = t_0$; otherwise, if the t_0-constraint for pattern i is inactive, then $\mu_0^i = 0$.

The split transformation is now described in detail. Suppose that consecutive completion epochs occur at the end of stages k and n with $k < n \leq J$. Otherwise, if k is the index of the final completion epoch, then set $n = J + 1$. At index k, there are $n - k$ simultaneous completions with jobs $k, \cdots, n-1$ completing. At index $n \leq J$, there can be a single completion or multiple simultaneous completions.

The split transformation introduces an index l, with $k < l < n$, such that jobs $k, \cdots, l-1$ complete at index k and jobs $l, \cdots, n-1$ complete at index l. Note that in pattern 1, $\tau_l = 0$. This is illustrated in Tables 2 and 3 for the two circumstances in which a split may occur (the index values used are illustrative). In the tables, a "1" at index j indicates that $\tau_j > 0$ and a "0" indicates $\tau_j = 0$; whereby the split transformation results in the insertion of an additional "1" at index l. Denote by M^1 and M^2 the index of the final completion epoch in pattern 1 and pattern 2, respectively. The two circumstances in which a split can occur are the following: In an *end-split transformation* (illustrated in Table 2), index k is the index of the final completion epoch of pattern 1. In an *internal-split transformation* (illustrated in Table 3), index k occurs before the final completion epoch. Observe that in an end-split transformation, the index of the final completion epoch increases from $M^1 = k$ to index $M^2 = l$. In an internal-split transformation, the index of the final completion epoch is unchanged, $M^1 = M^2 = n$.

7.2 Change in makespan and total flow time

This section derives expressions for the change in makespan and total flow time for a single value of μ. Suppose that before the transformation the makespan and total flow time are $T^1(\mu)$ and $S^1(\mu)$, respectively, and after the transformation the makespan and total flow time are $T^2(\mu)$ and $S^2(\mu)$, respectively. Expressions for the change in makespan, $T^2(\mu) - T^1(\mu)$, and total flow time, $S^2(\mu) - S^1(\mu)$, are derived in the following lemma. (From now on, the function argument μ will be deleted, unless required.)

Lemma 7.1 *Define*

$$a_k = \frac{\widetilde{x}_{k:n-1}}{\sigma^{cp}_{k:n-1}}, \quad a_l = \frac{\widetilde{x}_{k:n-1}}{\sigma^{cp}_{k:n-1}},$$
$$b_k = \frac{\widetilde{x}_{k:l-1}}{\sigma^{cp}_{k:l-1}}, \qquad b_l = \frac{\widetilde{x}_{l:n-1}}{\sigma^{cp}_{l:n-1}},$$

and

$$u_k = \alpha_k^{cp} - \alpha_l^{cp}, \quad u_l = \alpha_l^{cp} - \alpha_n^{cp},$$
$$v_k = (J-k+1)\alpha_k^{cp} - (J-l+1)\alpha_l^{cp},$$
$$v_l = (J-l+1)\alpha_l^{cp} - (J-n+1)\alpha_n^{cp}.$$

Define the vectors $\boldsymbol{a} = (a_k, a_l)$, $\boldsymbol{b} = (b_k, b_l)$, $\boldsymbol{u} = (u_k, u_l)$ *and* $\boldsymbol{v} = (v_k, v_l)$.
(i)

$$T^2 - T^1 = (b_k u_k + b_l u_l) - (a_k u_k + a_l u_l) = (\boldsymbol{b} - \boldsymbol{a}) \cdot \boldsymbol{u} \tag{54}$$
$$= \widetilde{x}_{k:l-1} \frac{\alpha_k^{cp} - \alpha_l^{cp}}{\sigma^{cp}_{k:l-1}} + \widetilde{x}_{l:n-1} \frac{\alpha_l^{cp} - \alpha_n^{cp}}{\sigma^{cp}_{l:n-1}} - \widetilde{x}_{k:n-1} \frac{\alpha_k^{cp} - \alpha_n^{cp}}{\sigma^{cp}_{k:n-1}}. \tag{55}$$

(ii) (a)

$$S^2 - S^1 = (b_k v_k + b_l v_l) - (a_k v_k + a_l v_l) = (\boldsymbol{b} - \boldsymbol{a}) \cdot \boldsymbol{v}. \tag{56}$$
$$= \widetilde{x}_{k:l-1} \frac{(J-k+1)\alpha_k^{cp} - (J-l+1)\alpha_l^{cp}}{\sigma^{cp}_{k:l-1}}$$
$$+ \widetilde{x}_{l:n-1} \frac{(J-l+1)\alpha_l^{cp} - (J-n+1)\alpha_n^{cp}}{\sigma^{cp}_{l:n-1}}$$
$$- \widetilde{x}_{k:n-1} \frac{(J-k+1)\alpha_k^{cp} - (J-n+1)\alpha_n^{cp}}{\sigma^{cp}_{k:n-1}}. \tag{57}$$

(ii) (b) An alternative expression for $S^2 - S^1$ *is given by*

$$S^2 - S^1 = (\widehat{S}^2 - \widehat{S}^1) - \mu(T^2 - T^1), \tag{58}$$

where

$$\widehat{S}^1 = \widetilde{x}_{k:n-1} \frac{\alpha_k^c - \alpha_n^c}{\sigma^{cp}_{k:n-1}} = \widetilde{x}_{k:n-1} (\alpha_k^c - \alpha_n^c)^{1-p}, \tag{59}$$
$$\widehat{S}^2 = \widetilde{x}_{k:l-1} \frac{\alpha_k^c - \alpha_l^c}{\sigma^{cp}_{k:l-1}} + \widetilde{x}_{l:n-1} \frac{\alpha_l^c - \alpha_n^c}{\sigma^{cp}_{l:n-1}}$$
$$= \widetilde{x}_{k:l-1} (\alpha_k^c - \alpha_l^c)^{1-p} + \widetilde{x}_{l:n-1} (\alpha_l^c - \alpha_n^c)^{1-p}. \tag{60}$$

Proof See Appendix B. □

8 Optimality for scenarios

The first part of proving Theorem 4.1(iii) is to consider patterns that are only scenarios. Recall that scenario M^* is the scenario with the largest value of M that is τ_n-viable. The proof involves showing that scenario M^* has the smallest total flow time. That is, $M^* = \arg\max_M\{\text{Scenario } M \text{ is } \tau_n\text{-viable}\}$.

The optimality of scenarios is demonstrated by using end-split transformations. In particular, scenario M can be obtained from scenario $M-1$ by using an end-split transformation with $k = M-1$ and $l = M$. The sequence of scenarios $1, \cdots, J$, can be obtained by starting at scenario 1 and performing successive end-split transformations at the next index after the current final completion epoch index.

8.1 Intervals

The effect of the end-split transformation depends on the interval in which t_0 is contained. This section determines the key intervals that define the structure of the sequence of scenarios $1, \cdots, J$. The following ordering determines t_0-viability.

Lemma 8.1 *(i) For* $M > 1$, $T^M(\infty) < T^{M-1}(\infty)$.
(ii)

$$T^J(\infty) < \cdots < T^1(\infty). \tag{61}$$

Proof (i) This follows since $T^M(\infty) = \widetilde{x}_{M:J} < T^{M-1}(\infty) = \widetilde{x}_{M-1:J}$. The inequalities are strict since job sizes are assumed to be nonzero.
(ii) This follows immediately.

□

By Proposition 6.6(iii), if scenario M is t_0-viable, then τ_n-viability requires $T^M_\triangle < t_0$. To obtain an ordering for $T^M_\triangle$, we begin with Proposition 8.1 that determines the change $T^M(\mu) - T^{M-1}(\mu)$ for a given value of μ.

Proposition 8.1 *For an end-split transformation,*

$$T^M(\mu) - T^{M-1}(\mu) \quad \begin{cases} > 0, \ \mu < \mu^M_\triangle \\ = 0, \ \mu = \mu^M_\triangle \\ < 0, \ \mu > \mu^M_\triangle \end{cases}.$$

Proof Set $k = M-1$ and $l = M$ in Proposition C.1. (See Appendix C.1). □

The intuition behind the case of zero change is that if $\mu = \mu^M_\triangle$ then $\tau^M_M = 0$, in which case scenario M becomes identical to scenario $M-1$ and they have the same makespan (and, as shown below in Proposition 8.2, the total flow time is the same). We also require the following lemma.

Lemma 8.2 *For* $M > 1$, $\mu^M_\triangle < \mu^{M-1}_\triangle$.

Proof Set $k = M - 1$ and $l = M$ in Lemma C.1. □

Using Proposition 8.1 and Lemma 8.2, the ordering of $T_\triangle^M$ can be obtained.

Lemma 8.3 *(i) For $M > 1$, $T_\triangle^M < T_\triangle^{M-1}$.*
(ii)

$$T_\triangle^1 < \cdots T_\triangle^J. \tag{62}$$

(iii)

$$T^J(\infty) < \cdots < T^1(\infty) = T_\triangle^1 < \cdots T_\triangle^J. \tag{63}$$

Proof (i) Since T^{M-1} is decreasing and, according to Lemma 8.2, $\mu_\triangle^M < \mu_\triangle^{M-1}$, $T_\triangle^{M-1} = T^{M-1}(\mu_\triangle^{M-1}) < T^{M-1}(\mu_\triangle^M)$. By Proposition 8.1, $T^{M-1}(\mu_\triangle^M) = T_\triangle^M$, and the result follows.
(ii) This follows immediately.
(iii) Since $T^1(\infty) = T_\triangle^1$, combining (61) with (62) gives the result.

□

8.2 Application of the t_0-constraint

The change in t_0-viability from one scenario to another is given by the following lemma.

Lemma 8.4 *(i) If scenario $M - 1$ is t_0-viable, then scenario M is t_0-viable.*
(ii) If scenario M is not t_0-viable, then scenario $M - 1$ is not t_0-viable.
(iii) If $T^M(\infty) < t_0 \leq T^{M-1}(\infty)$, then scenario M is t_0-viable but scenario $M - 1$ is not t_0-viable.

Proof (i) If scenario $M - 1$ is t_0-viable, then $T^{M-1}(\infty) < t_0$. Since $T^M(\infty) < T^{M-1}(\infty)$, $T^M(\infty) < t_0$. Hence, scenario M is t_0-viable.
(ii) and (iii) follow similar lines to (i).

□

Next, consider the τ_n-constraints. The change in τ_n-viability from one scenario to another is given by the following lemma.

Lemma 8.5 *Suppose that scenarios M and $M - 1$ are t_0-viable.*

(i) If scenario M is τ_n-viable, then scenario $M - 1$ is τ_n-viable.
(ii) If scenario $M - 1$ is not τ_n-viable, then scenario M is not τ_n-viable.
(iii) If scenario $M - 1$ is τ_n-viable, then either (a) scenario M is τ_n-viable (if $T_\triangle^2 < t_0$) or (b) scenario M is not τ_n-viable (if $T_\triangle^{M-1} < t_0 < T_\triangle^M$).

Proof (i) $T_\triangle^M < t_0$ and $T_\triangle^{M-1} < T_\triangle^M$ imply $T_\triangle^{M-1} < t_0$.
(ii) $t_0 < T_\triangle^{M-1}$ and $T_\triangle^{M-1} < T_\triangle^M$ implies $t_0 < T_\triangle^M$.

Table 4 Different intervals for t_0

	t_0 interval	t_0-viable		τ_n-viable	
		$M-1$	M	$M-1$	M
(A)	$T_\triangle^M < t_0$	✓	✓	✓	✓
(B_1)	$T_\triangle^{M-1} < t_0 \le T_\triangle^M$	✓	✓	✓	×
(B_2)	$T^{M-1}(\infty) < t_0 \le T_\triangle^{M-1}$	✓	✓	×	×
(C)	$T^M(\infty) < t_0 \le T^{M-1}(\infty)$	×	✓	·	×
(D)	$t_0 \le T^M(\infty)$	×	×	·	·

(iii) $T_\triangle^{M-1} < t_0$ divides into two cases: (a) $T_\triangle^{M-1} < T_\triangle^M < t_0$ and (b) $T_\triangle^{M-1} < t_0 < T_\triangle^M$. In case (a), scenario M is τ_n-viable, and in case (b), scenario M is not τ_n-viable.

□

Table 4 summarizes how τ_0 -viability and τ_n -viability depend on the interval to which t_0 belongs. In the table '✓'='property is true, '×'='property is false, and '·' means that τ_n-viability cannot be determined as the t_0-constraint for the scenario is not satisfied. (See also Fig. 6.)

When the t_0-constraint is included, different patterns typically require different values of μ to satisfy the t_0-constraint. The following lemma shows how the relationship between μ_0^{M-1}, μ_0^M and $\mu_\triangle^M$ depends on the τ_n-viability of Scenario M.

Lemma 8.6 *For an end-split transformation:*

(i) If scenario M is τ_M-viable then $\mu_0^{M-1} < \mu_0^M < \mu_\triangle^M$.
(ii) If scenario M is not τ_M-viable then $\mu_\triangle^M < \mu_0^M < \mu_0^{M-1}$.

Proof (i) Since scenario M is τ_M-viable, $T_\triangle^M < t_0$, which gives $\mu_0^M < \mu_\triangle^M$. From Proposition 8.1, $T^{M-1}(\mu_0^M) < T^M(\mu_0^M) = t_0$. However, since $t_0 = T^{M-1}(\mu_0^{M-1})$, we obtain $T^{M-1}(\mu_0^M) < T^{M-1}(\mu_0^{M-1})$, which implies $\mu_0^{M-1} < \mu_0^M$, since T^{M-1} is a decreasing function.

(ii) The proof is the same as in (i) except that the inequalities are reversed.

□

8.3 Multiple scenarios

We now consider the sequence of scenarios $1, \cdots, J$. Using Lemma 8.4, there is a smallest value of M that is t_0-viable. Denote this value by M'. If no scenario is t_0-viable, then set $M' = J + 1$. M' is determined by the condition that $T^M(\infty) < t_0 \le T^{M-1}(\infty)$. It follows that scenarios $1, \cdots, M' - 1$ are not t_0-viable and scenarios $M', \cdots, J$ are t_0-viable.

Using Lemma 8.5, there is a largest value of M that is τ_n-viable. Denote this value by M^*. Since scenario 1 is τ_n-viable, we have $1 \le M^*$. M^* is determined by the

condition $T_{\triangle}^{M^*} < t_0 \leq T_{\triangle}^{M^*+1}$. It follows that scenarios $1, \cdots, M^*$ are τ_n-viable and scenarios $M^*+1, \cdots, J$ are not τ_n-viable.

It is now established that all scenarios are t_0-viable or no scenarios are t_0-viable.

Lemma 8.7 *(i) If $T^1(\infty) = \widetilde{x}_{1:J} \leq t_0$, then all scenarios are t_0-viable.*
(ii) If $t_0 < T^1(\infty)$ there are no scenarios that are both t_0-viable and τ_n-viable.

Proof t_0-viability of scenario M requires $T^M(\infty) < t_0$.

(i) The result follows immediately from the ordering given in (61).
(ii) If $t_0 < T^1(\infty)$ then (a) $t_0 \leq T^J(\infty)$ or (b) $T^J(\infty) < t_0 < T^1(\infty)$. In case (a), no scenario is t_0-viable. In case (b), M', as defined above, satisfies $2 \leq M' \leq J$. If $M < M'$, then scenario M is not t_0-viable, and the proof is complete. If $M' \leq M$ then scenario M is t_0-viable. However, scenario M is not τ_n-viable because the τ_n-viability of scenario M requires $T_{\triangle}^M < t_0$, but, using the ordering in (63), $t_0 < T^1(\infty) < T_{\triangle}^M$. □

Remark 2 Note that in Lemma 8.7(i), even though all scenarios are t_0-viable, it is possible to have some scenarios τ_n-viable and others not τ_n-viable.

The following lemma gives the ordering of $\mu_{\triangle}$ and μ_0 values for all scenarios.

Lemma 8.8 *(i)*

$$\mu_{\triangle}^J < \cdots < \mu_{\triangle}^1.$$

(ii) (a)

$$\mu_0^1 < \cdots < \mu_0^{M^*},$$

(b)

$$\mu_0^{M^*} > \mu_0^{M^*+1} > \cdots > \mu_0^J.$$

Proof (i) This follows from Lemma 8.2.
(ii) (a) For $M \leq M^*$, scenario M is τ_n-viable and $\mu_0^M < \mu_{\triangle}^M$. Using Lemma 8.6, $\mu_0^{M-1} < \mu_0^M < \mu_{\triangle}^M$.
(b) For $M > M^*$, scenario M is not τ_n-viable, and $\mu_{\triangle}^M < \mu_0^M$. Using Lemma 8.6, $\mu_0^{M-1} > \mu_0^M > \mu_{\triangle}^M$.

□

8.4 Total flow time

This section completes the proof of Theorem 4.1(iii) for the case where patterns are scenarios. It is assumed that $T^1(\infty) < t_0$ so that all scenarios are t_0-viable, otherwise no viable scenarios exist. First, the change in total flow time for a single value of μ is determined.

Proposition 8.2 *For an end-split transformation:*

$$S^M(\mu) - S^{M-1}(\mu) \quad \begin{cases} < 0, \ \mu < \mu_{\triangle}^M \\ = 0, \ \mu = \mu_{\triangle}^M \\ > 0, \ \mu > \mu_{\triangle}^M \end{cases}.$$

Proof Set $k = M - 1$ and $l = M$ in Proposition C.2. (see Appendix C.2). □

Proposition 8.3 introduces the t_0 constraint and shows that the change in total flow time from scenario $M - 1$ to M depends on whether scenario M is τ_n-viable or not. Recall that $S_0^{M-1} \equiv S^{M-1}(\mu_0^{M-1})$ and $S_0^M \equiv S^M(\mu_0^M)$.

Proposition 8.3 *(i) If scenario M is τ_n-viable, then $S_0^M < S_0^{M-1}$.*
(ii) If scenario M is not τ_n-viable, then $S_0^{M-1} < S_0^M$.

Proof (i) In this case, $T_{\triangle}^M < t_0$. Lemma C.3(i) then proves that $S_0^M < S_0^{M-1}$.
(ii) In this case, $T^{M-1}(\infty) < t_0 \leq T_{\triangle}^M(\infty)$. Lemma C.3(ii) then proves that $S_0^M > S_0^{M-1}$.

□

We now prove that the optimal scenario is scenario M^*.

Proposition 8.4 *Suppose $\widetilde{x}_{1:J} \leq t_0$. (i) For $M \leq M^*$, scenario M is τ_n-viable and $S_0^{M^*} \leq S_0^M$.*
(ii) For $M^* < M$, scenario M is not τ_n-viable and $S_0^{M^*} < S_0^M$.

Proof Consider scenario M.

(i) If $M \leq M^*$, then scenario M is τ_n-viable. By Proposition 8.3(i), $S_0^M < S_0^{M-1}$.
(ii) If $M > M^*$, then scenario M is not τ_n-viable. By Proposition 8.3(ii), $S_0^{M+1} > S_0^M$.
(i) and (ii) give the following ordering:

$$S_0^1 > \cdots > S_0^{M^*-1} > S_0^{M^*} < S_0^{M^*+1} < S_0^J. \tag{64}$$

It follows that scenario M^* has the smallest value of S_0. □

Remark 3 Proposition 8.4(ii) includes scenarios that are not τ_n-viable. Such scenarios will be used in the proof of optimality when patterns other than scenarios are included.

Remark 4 As the above proofs use the results for general end-split transformations, the optimality of the scenario sequence $1, \cdots, J$ extends to any sequence of patterns constructed using end-split transformations. This observation will be used in the proof of Lemma 9.3. If the patterns are ordered according to the order of application of the end-split transformation, then M^* is the last pattern in the sequence that is τ_n-viable and it will have the smallest total flow time for the given pattern sequence.

Table 5 Pattern numbers for $J = 5$

0	10000
1	11000
2	10100
3	11100
4	10010
...	...
14	10111
15	11111

Table 6 Pattern numbers of scenarios for $J = 5$

Scenario M	Pattern	Pattern number
1	10000	0
2	11000	1
3	11100	3
4	11110	7
5	11111	15

9 Optimality for all patterns

The second part of the proof of Theorem 4.1(iii) extends the proof of scenarios to include all patterns. Recall that the tuple $(b_1, \cdots, b_J)$ represents a completion pattern, where bit b_n equals 0 or 1 depending on whether $\tau_n = 0$ or $\tau_n > 0$, respectively. Since $\tau_1 > 0$, bit $b_1 = 1$ always. The *pattern number* is the numeric value of $(b_2, \cdots, b_J)$ interpreted as a binary number where the most significant bit is b_J and the bit weights are $2^0, 2^1, \cdots, 2^{J-1}$. The range of completion pattern numbers is 0 to $2^{J-1} - 1$. An example of the completion numbers for $J = 5$ is shown in Table 5. In this numbering scheme, all patterns with the same index for the final completion epoch are grouped sequentially. The pattern number for scenario M is $2^{M-1} - 1$. Table 6 lists all possible scenarios for $J = 5$. The notation $\widetilde{M} \equiv 2^{M-1} - 1$ is used to specify the pattern number for scenario M.

Patterns cannot be totally ordered using internal-split transformations, but can be partially ordered. For example, the patterns $N^1 = 100101$ and $N^2 = 101001$ are each derived from pattern 100001 using an internal-split transformation. However, there is no sequence of internal-split transformations that can transform N^1 to N^2 and vice versa. The approach will be as follows. Suppose that pattern N has the last completion epoch index M. We generate a pattern sequence using internal-split transformations that terminates in scenario M. The relationship between N and M is that of $\widetilde{M-1} < N < \widetilde{M}$. Suppose that there are m internal-split transformations that produce this pattern sequence. Denote the pattern sequence by $\mathcal{N} = (N_0, N_1 . \cdots, N_m)$ ($N_0 = N$ and $N_m = \widetilde{M}$). The last completion epoch index for each pattern in the sequence is M, since an internal-split transformation does not change that value. Having determined scenario M, Scenarios M, $M - 1$ and M^* can be compared using the total ordering results for the scenarios. It will be shown that if pattern N is τ_n-viable, then it will

have a larger total flow time than scenario M^*. Although it may be possible for pattern N to have $S_0^N < S_0^{\widetilde{M}^*}$, pattern N cannot be τ_n-viable.

9.1 Intervals

In this section, key values are presented that define the intervals for makespan that allow the viability of the patterns in $\mathscr{N}$ to be determined.

Lemma 9.1 *(i) For* $0 \le i, j \le m$, $T^{N_i}(\infty) = T^{N_j}(\infty)$.
(ii)

$$T^1(\infty) = \cdots = T^J(\infty). \tag{65}$$

Proof (i) This follows since $T^{N^i}(\infty) = T^{N^j}(\infty) = \widetilde{x}_{M:J}$.
(ii) This follows immediately.

□

For the following Lemmas 9.2 and 9.3, pattern 1 is the pattern before the internal-split transformation and pattern 2 is the pattern after the internal-split transformation.

Lemma 9.2 *(i) If* $n < M$ *then* $\mu_\triangle^2 = \mu_\triangle^1$.
(ii) If $n = M$, *then* $\mu_\triangle^2 < \mu_\triangle^1$.

Proof See Appendix C.1. □

We prove the following.

Lemma 9.3 *(i) If* $n < M$, *then both* $T_\triangle^1 < T_\triangle^2$ *and* $T_\triangle^2 < T_\triangle^1$ *are possible.*
(ii) If $n = M$, *then* $T_\triangle^1 < T_\triangle^2$.

Proof See Appendix C.1. □

9.2 Application of the t_0-constraint

In this section, a given value of t_0 is applied. Lemmas 9.4 and 9.5, determine t_0-viability and τ_n-viability, respectively. In these lemmas, pattern 1 is the pattern before the internal-split transformation and pattern 2 is the pattern after the internal-split transformation.

Lemma 9.4 *(i) For an internal-split transformation and a given value of* t_0, *the* t_0*-viability of pattern 2 is the same as the* t_0*-viability of pattern 1.*
(ii) The t_0*-viability of patterns 1 and 2 is the same as for Scenario* $M = M^1 = M^2$.

Proof (i) Since $M_1 = M_2$ then $t_0 > T^1(\infty) = \widetilde{x}_{M^1:J}$ if and only if $t_0 > T^2(\infty) = \widetilde{x}_{M^2:J}$. Thus, the t_0-viability of patterns 1 and 2 is the same.
(ii) For both pattern 1 and pattern 2, Scenario M can be reached by a sequence of internal-split transformations. Thus, they all have the same final epoch index ($M = M^1 = M^2$) which means that the t_0-viability is the same. □

Table 7 τ_n-viability for the internal-split transformation

	t_0 interval	1 viable	2 viable
(a)	$t_0 < T^1_\Delta, T^2_\Delta$	×	×
(b)	$T^1_\Delta < t_0 \le T^2_\Delta$	✓	×
(c)	$T^2_\Delta < t_0 \le T^1_\Delta$	×	✓
(d)	$T^1_\Delta, T^2_\Delta < t_0$	✓	✓

Lemma 9.5 *For an internal-split transformation and a given value of t_0:*

(i) If $n < M$, then any combination of τ_n-viability can occur with patterns 1 and 2.
(ii) If $n = M$, then any combination of τ_n-viability can occur with patterns 1 and 2 except for the case where pattern 1 is τ_n-viable and pattern 2 is not τ_n-viable.

Proof (i) From Lemma 9.3 both $T^1_\Delta < T^2_\Delta$ and $T^2_\Delta < T^1_\Delta$ are possible. This gives the four combinations (a)–(d) shown in Table 7.
(ii) From Lemma 9.3, only $T^1_\Delta < T^2_\Delta$ is possible. This gives the same combinations as in (i), except that $T^2_\Delta < t_0 < T^1_\Delta$, is not possible. Only combinations (a), (b), and (d) can occur in Table 7. □

9.3 Pattern sequence

This section provides a condition for all patterns to be t_0-viable.

Proposition 9.1 *(i) If $\widetilde{x}_{1:J} \le t_0$, then all patterns are t_0-viable.*
(ii) If $t_0 < \widetilde{x}_{1:J}$, then no patterns are both t_0-viable and τ_n-viable.

Proof (i) The assumption $\widetilde{x}_{1:J} \le t_0$ implies that scenario M is t_0-viable. Since $T^N(\infty) = T^{\widetilde{M^*}} = \widetilde{x}_{M:J}$ then $T^N(\infty) < t_0$ if and only if $T^{\widetilde{M^*}} < t_0$. Thus, pattern N is also t_0-viable.
(ii) The proof is similar to Lemma 8.7(ii).

□

9.4 Total flow time

As in the case of scenarios, it is assumed that $\widetilde{x}_{1:J} \le t_0$. For Propositions 9.2 and 9.3 following, pattern 1 is the pattern before the internal-split transformation and pattern 2 is the pattern after the internal-split transformation.

Numerical studies show that makespan may decrease or increase for an internal-split transformation. However, Sect. 9.6 shows that makespan will increase for two broad cases. The following proposition proves that regardless of whether makespan decreases or increases total flow time always decreases for a given value of μ.

Proposition 9.2 *For an internal-split transformation $S^2(\mu) < S^1(\mu)$.*

Proof See Appendix D.2. □

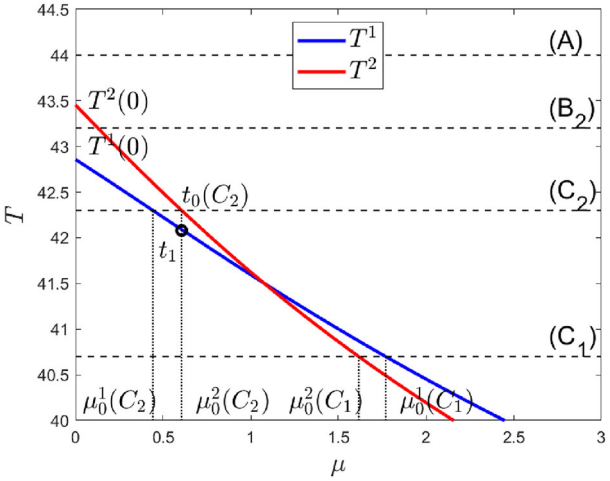

Fig. 4 Internal-split transformation: T

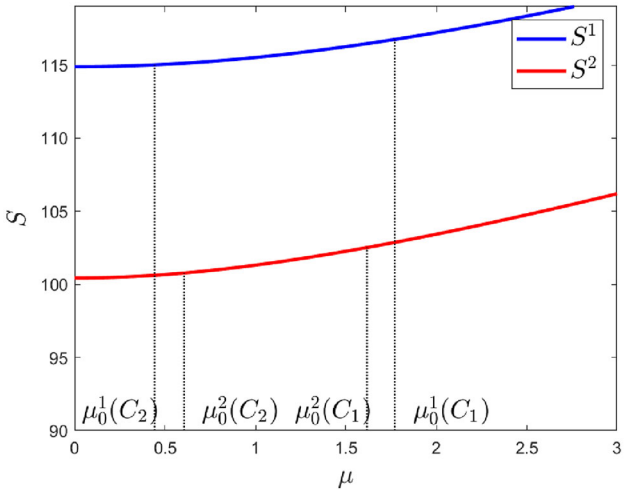

Fig. 5 Internal-split transformation: S

Proposition 9.3 *For an internal-split transformation,* $S_0^2 < S_0^1$.

Proof Different intervals for t_0 must be considered (see Fig. 4).

(A) $t_0 > T^1(0)$ and $t_0 > T^2(0)$.

In this case, $\mu_1 = \mu_0^2 = 0$, and so, by Proposition D.1, $S_0^2 = S^2(0) < S^1(0) = S_0^1$.

(B$_1$) $T^2(0) < t_0 < T^1(0)$.

In this case, $\mu_0^1 > 0$ and $\mu_0^2 = 0$. (This case is not shown in Fig. 4.) Again, by Proposition D.1, $S_0^2 = S^2(0) < S^1(0)$. Since S^1 is increasing, $S^1(0) < S_0^1$, giving $S_0^2 < S_0^1$.

(B$_2$) $T^1(0) < t_0 < T^2(0)$.

In this case, $\mu_0^1 = 0$ and $0 < \mu_0^2$, and so, by Lemma D.4(ii), $S_0^2 < S^1(0)$.

(C) $T_0^1 = T_0^2 = t_0$.

In this case, $0 < \mu_0^1$, $0 < \mu_0^2$. It is possible to have $T^1(0) < T^2(0)$ and yet $T^2(\mu_0^1) < T^1(\mu_0^1) = t_0$; and, similarly if $T^2(0) < T^1(0)$, then it is possible that $T^2(\mu_0^1) > T^1(\mu_0^1) = t_0$. This gives the following two subcases:

- (C_1) $T^2(\mu_0^1) < T^1(\mu_0^1) = t_0$. In this case, $\mu_0^2 < \mu_0^1$, and so, by Proposition D.1, $S^2(\mu_0^1) < S_0^1$. Since S^2 is increasing, $S_0^2 < S^2(\mu_0^1)$, giving $S_0^2 < S_0^1$.
- (C_2) $t_0 = T^1(\mu_0^1) \le T^2(\mu_0^1)$. In this case, $\mu_0^1 < \mu_0^2$, and Lemma D.4(i) shows $S_0^2 \le S_0^1$.

(See Fig. 5 for an example of (C_1) and of (C_2) for the total flow time.) □

Remark 5 Depending on the position of t_0 relative to $T_\triangle^1$ and $T_\triangle^2$, it is possible for one or both of pattern 1 and pattern 2 to be not τ_n-viable (see Lemma 9.3). However, $S_0^2 < S_0^1$ will always be true.

9.4.1 Completion of proof of Theorem 4.1(iii)

Proof Consider the pattern sequence $\mathcal{N}$ generated by pattern N that terminates with scenario M, such that $\widetilde{M-1} < N < \widetilde{M}$ and we have defined $M^* = \arg\max_M\{\text{Scenario } M \text{ is } \tau_n\text{-viable}\}$. The following cases are considered.

Scenario *M* is viable

If scenario M is τ_n-viable, then $M \le M^*$, and $S_0^{\widetilde{M^*}} \le S_0^{\widetilde{M}}$. At split i in the pattern sequence, the τ_n-viability can change, and the change in makespan, $T_0^{N_i} - T_0^{N_{i-1}}$, can be positive, negative, or zero. However, by Proposition 9.3, $S_0^{N_i} < S_0^{N_{i-1}}$. Therefore, $S_0^{\widetilde{M}} < S_0^N$. The result then follows since $S_0^{\widetilde{M^*}} < S_0^{\widetilde{M}}$.

Scenario *M* − 1 is viable but scenario *M* is not viable

Now, suppose that scenario $M-1$ is τ_n-viable, but scenario M is not τ_n-viable (see the example in Table 8). In this case $M^* = M-1$. Since an internal-split transformation reduces the total flow time (regardless of changes in τ_n-viability), scenario M satisfies $S_0^{\widetilde{M}} < S_0^N$. Next, since scenario M is not τ_n-viable, by Proposition 8.4, $S_0^{\widetilde{M^*}} < S_0^{\widetilde{M}}$. Combining the inequalities gives $S_0^{\widetilde{M^*}} < S_0^N$.

Scenario *M* − 1 is not viable

It may also happen that pattern N is τ_n-viable with $\widetilde{M-1} < N < \widetilde{M}$ but neither scenario $M-1$ nor scenario M are τ_n-viable. In this case, $M^* < M-1$. As in Sect. 9.4.1 $S_0^{\widetilde{M}} < S_0^N$. Since scenario $M-1$ is not viable, by Proposition 8.4, $S_0^{\widetilde{M^*}} < S_0^{\widetilde{M-1}} < S_0^{\widetilde{M}}$. Combining the two preceding inequalities gives $S_0^{\widetilde{M^*}} < S_0^N$. □

Table 8 Example ($J = 8$, $M = 5$)

Pattern index	Example	Viability
$\widetilde{M-1} = M^*$	11111000	Viable
...	...	
N	10011100	Viable
...	...	
$\widetilde{M}$	11111100	Not viable

Table 9 Example listing all completion patterns ($J = 4$)

N	Pattern	Scenario	T	S	τ_M	μ_0	$\mu_\triangle$
0	1000	1	27.3281	109.312	27.3281	0	∞
1	1100	2	27.9293	94.1171	17.6	9.70373	54.7717
2	1010		27.9293	101.407	5.15516	10.7215	15.9349
3	1110	3	27.9293	94.1049	−0.32374	9.0112	8.79891
4	1001		27.9293	105.124	2.19783	6.27307	7.70346
5	1101		27.9293	93.7089	−0.823933	5.61709	5.21483
6	1011		27.9293	98.6287	−5.6523	6.21692	3.76684
7	1111	4	27.9293	92.3566	−4.54055	5.70992	3.76684

9.5 Example

Table 9 gives an example listing all patterns for $J = 4$ ($p = 0.7$, $x = (3.6154, 10.1050, 12.5868, 13.6927)$). In this example, the minimum makespan is 27.3281 and t_0 is set to 27.9293. The minimum value of the total flow time is 94.1171, which occurs for scenario 2. Other patterns have lower values for S ($N = 3, 5$), however, these are not viable since $\tau_M < 0$.

9.6 Equal job sizes

Equal job sizes have practical application. They also provide a limiting case for analysis. For an internal-split transformation with equal job sizes, and a given value of μ, conditions for $T^2(\mu) - T^1(\mu) > 0$ will be given for two broad cases. Define $a = l - k$ and $b = n - l$. The two cases are as follows:

1. $p \leq 0.5$ and all possible a, b. This provides a total ordering between completion patterns, but does not cover the full range of p values. The result is given in Proposition D.1(i).
2. $a \geq b$ and $0 < p < 1$. This provides a partial order between the completion patterns for all values of p. This result is given in Proposition D.1(ii).

Although we proved the result for two broad cases, numerical studies show that it is always true.

We also prove, for an internal-split transformation, that equal job sizes produce the least reduction in total flow time, subject to a constraint on job sizes.

Proposition 9.4 *Consider an internal-split transformation. Suppose $\widetilde{x}_{k:n-1}$ is fixed at a given value. Then, the maximum value of the change in the total flow time $S^2(\mu) - S^1(\mu)$ occurs when jobs are of equal size.*

Proof See Appendix D.4.3. □

9.7 No completion deadline

The standard case, where there is no completion deadline constraint, such as given by [6], can be analyzed by letting $t_0 \to \infty$. In this case, for the minimal flow-time solution, jobs complete in SJF order, and $\mu_0 = 0$. From Lemma 8.7 all scenarios are t_0-viable and from Lemma 9.1 all patterns are t_0-viable. Further, all scenarios are τ_n-viable by using the fact that scenario 1 is τ_n-viable and using Lemma 8.5. Therefore, $M^* = J$, so that jobs are completed individually, and the minimum total flow time is obtained by scenario J.

9.8 Makespan minimization

Theorems 1 and 2 of [6] prove that if the optimization problem is to minimize the makespan, then the minimum makespan will be obtained when all jobs are completed at the same time and the minimum makespan value will be $\widetilde{x}_{1:J}$. These results are proved using an argument based on the reallocation of the cores and the convexity of the speedup function. The same results can also be obtained using the methods described in this paper. Consider any ordering of jobs and the associated completion pattern. Following Remark 1, the makespan minimization problem can be derived by using the weights $w_n = 1$ in the objective function $\sum_{n=1}^{J} w_n \tau_n$. By making the value of t_0 large, $\mu = 0$, which gives $\alpha_n = 1$. This gives $\widetilde{\sigma}_{M:J} = 1$ and $\sigma_{n:\overline{n}} = 0$ ($n < M$ and $\tau_n > 0$) (see Lemma A.6). However, for $M > 1$, inserting these values into (34) gives $\tau_M = -\infty$. Only in the case of $M = 1$ will $\tau_M > 0$ be possible. In this case, all jobs are completed at the same time and the makespan is $T = \tau_1 = \widetilde{x}_{1:J}$, which is the value given by [6].

10 Conclusion

This paper had the goal of optimizing the performance of a multicore system processing parallelizable jobs that had a power-law speedup function in the presence of a completion deadline. All jobs processed were assumed to be present at time 0. Closed-form expressions were derived for the optimal core allocations to minimize total flow time using the Lagrange multiplier technique for $0 < p < 1$. Equations predicting the total flow time value and makespan were obtained as finite sums that showed the dependence on job sizes. Compared to the unconstrained problem, similarities include preservation of the SJF ordering property and allocations only changing at completions. With a completion deadline, a significant difference was that simultaneous completions and makespan needed to be considered, leading to more complex tradeoffs. The scale-free property had to be modified to support the completion dead-

line, with the paper uncovering the dependence between that and the completion time constraint Lagrange multiplier. As a consequence, the size-dependence property of allocations was modified, and the universal quantization property was lost. Analytically, it was shown that the optimal completion pattern had the form of a scenario. This reduced the size of the search space of completion patterns from exponential to linear in the number of jobs. The optimal allocation for equal job sizes was shown to provide an upper bound on the change in total flow time and, under broad conditions, to increase makespan. The results will benefit current system operators with SLAs that express latency targets. Future work would examine the application of the paper's results to systems with online arrivals.

Appendix A Detailed solution of KKT conditions

This appendix derives the core allocations, stage durations, makespan and total flow time for the optimal solution in terms of the system primitives and μ.

A.1 Optimal core allocation—Proof of Proposition 6.1

The proof of Proposition 6.1 begins by using the KKT conditions to obtain expressions for λ_n.

Lemma A.1 *For $1 \leq n \leq J$ and $n \leq j$,*

$$\lambda_n = p\sigma_j \tau_n \theta_{j,n}^{p-1} = p\sigma_j \tau_n \frac{s_{j,n}}{\theta_{j,n}}. \tag{A1}$$

Proof This follows immediately from the KKT conditions (20). □

The next lemma uses the previous result to express the core allocations, $\theta_{j,n}$, in terms of σ_j and α_n.

Lemma A.2 *Given n ($1 \leq n \leq J$) such that $\tau_n > 0$ and $n \leq j$,*

$$\theta_{j,n} = \frac{\sigma_j^c \psi_n^c}{\alpha_n^c \psi_n^{cp}} = \frac{\sigma_j^c}{\alpha_n^c} \psi_n.$$

Proof Applying (A1) for $n \leq j$, $j' \leq J$, gives

$$p\sigma_j \tau_n \frac{s_{j,n}}{\theta_{j,n}} = p\sigma_{j'} \tau_n \frac{s_{j',n}}{\theta_{j',n}}.$$

Since $\tau_n > 0$, τ_n divided out to give

$$\sigma_j \frac{s_{j,n}}{\theta_{j,n}} = \sigma_{j'} \frac{s_{j',n}}{\theta_{j',n}}.$$

Inserting this into (21), in which j has been replaced by j', gives the following, noting that the complementary slackness condition (25) gives $\nu_n = 0$.

$$(J-n+1)+\mu = \sum_{j'=n}^{J} \sigma_{j'} s_{j',n} = \sum_{j'=n}^{J} \sigma_j s_{j,n} \frac{\theta_{j',n}}{\theta_{j,n}}$$
$$= \sigma_j \frac{s_{j,n}}{\theta_{j,n}} \sum_{j'=n}^{J} \theta_{j',n} = \sigma_j \frac{s_{j,n}}{\theta_{j,n}} \psi_n = \frac{\sigma_j}{\theta_{j,n}^{1-p}} \psi_n. \tag{A2}$$

This equation determines how allocations are distributed across multiple stages in the optimal solution. The active and inactive t_0-constraint can be considered in the same equation by setting $\mu > 0$ and $\mu = 0$, respectively. Using (A2), the optimal allocations are given by

$$\theta_{j,n} = \frac{\sigma_j^c \psi_n^c}{((J-n+1)+\mu)^c}. \tag{A3}$$

With α_n defined by (26), this becomes

$$\theta_{j,n} = \frac{\sigma_j^c \psi_n^c}{\alpha_n^c \psi_n^{cp}} = \frac{\sigma_j^c}{\alpha_n^c} \psi_n.$$

□

The proof of Proposition 6.1 is completed by determining the values of σ_j. This final step begins by determining the relationship between σ_j and α_n.

Lemma A.3 *If $\tau_n > 0$, then*

$$\sum_{j=n}^{J} \sigma_j^c = \alpha_n^c. \tag{A4}$$

Proof An expression for σ_n is obtained by observing, for n ($\tau_n > 0$), that

$$\psi_n = \sum_{j=n}^{J} \theta_{j,n} = \sum_{j=n}^{J} \frac{\sigma_j^c \psi_n}{\alpha_n^c} = \frac{\psi_n}{\alpha_n^c} \sum_{j=n}^{J} \sigma_j^c,$$

where Lemma A.2 has been used for the second equality. This gives

$$\sum_{j=n}^{J} \sigma_j^c = \alpha_n^c.$$

□

There are two possibilities that are considered: that job j is completed on its own, or that multiple simultaneous jobs are completed at the same time as job j.

A.1.1 Single job completion

Lemma A.4 *For a single job completion at the end of stage n:*

$$\sigma_n = \begin{cases} (\alpha_n^c - \alpha_{n+1}^c)^{1/c} & n < J, \\ \alpha_J & n = J. \end{cases}$$

Proof Consider $n < J$ first. Since job n completes individually, $\tau_n > 0$ and $\tau_{n+1} > 0$. Applying Lemma A.3 for both n and $n+1$ gives

$$\sigma_n^c = \sum_{n'=n}^{J} \sigma_{n'}^c - \sum_{n'=n+1}^{J} \sigma_n^c = \alpha_n^c - \alpha_{n+1}^c,$$

giving, $\sigma_n = (\alpha_n^c - \alpha_{n+1}^c)^{1/c}$. If $n = J$, then the sum in (A4) consists of only one term, to give $\sigma_J = \alpha_J$. □

A.1.2 Multiple simultaneous completions

For the case of multiple simultaneous completions, the following lemma is first required.

Lemma A.5 *For $n = 1, \cdots, J$,*

$$p \frac{x_n}{\sigma_n^{cp}} = \sum_{j'=1, \tau_j' > 0}^{n} \frac{\lambda_{j'} \psi_{j'}}{\alpha_{j'}^c}. \tag{A5}$$

Proof Starting with (15) gives, for $n = 1, \cdots, J$,

$$\begin{aligned} x_n &= \sum_{j'=1}^{n} s_{n,j'} \tau_{j'} = \sum_{j'=1, \tau_j' > 0}^{n} \frac{\lambda_{j'} \theta_{n,j'}}{p \sigma_n} \\ &= \frac{\sigma_n^{cp}}{p} \sum_{j'=1, \tau_j' > 0}^{n} \frac{\lambda_{j'} \psi_j'}{\alpha_{j'}^c}, \end{aligned} \tag{A6}$$

where Lemma A.1 has been used to obtain the second equality and Lemma A.2 has been used to obtain the third equality. This gives the Lemma result. □

Lemma A.6 *If jobs $\underline{n} \le n \le \overline{n}$ complete simultaneously then*

$$\sigma_n^c = (\alpha_{\underline{n}}^c - \alpha_{\overline{n}}^c)^p \left(\frac{x_n}{\widetilde{x}_{\underline{n}:\overline{n}-1}} \right)^{1/p}.$$

Proof Consider first the case where job n is the first job in the group of jobs that are completed simultaneously, that is, $\tau_n > 0$ and $\underline{n} = n$.

Lemma A.5 gives

$$p\frac{x_n}{\sigma_n^{cp}} = \sum_{j'=1,\tau_{j'}>0}^{n} \frac{\lambda_{j'}\psi_{j'}}{\alpha_{j'}^c}.$$

Replacing n by n' ($n < n' \le \overline{n}$) gives

$$p\frac{x_{n'}}{\sigma_{n'}^{cp}} = \sum_{j'=1,\tau_{j'}>0}^{n'} \frac{\lambda_{j'}\psi_{j'}}{\alpha_{j'}^c} = \sum_{j'=1,\tau_{j'}>0}^{n} \frac{\lambda_{j'}\psi_{j'}}{\alpha_{j'}^c},$$

where the second equality is obtained by using $\tau_{j'} = 0$. Hence,

$$p\frac{x_n}{\sigma_n^{cp}} = p\frac{x_{n'}}{\sigma_{n'}^{cp}}.$$

This gives

$$\sigma_n^c = \sigma_{n'}^c \left(\frac{x_n}{x_{n'}}\right)^{1/p}. \tag{A7}$$

For the case where $\overline{n} \le J$, applying Lemma A.3 gives

$$\sigma_n^c + \cdots + \sigma_{\overline{n}-1}^c = \sum_{n'=n}^{J} \sigma_{n'}^c - \sum_{n'=\overline{n}}^{J} \sigma_{n'}^c = \alpha_n^c - \alpha_{\overline{n}}^c.$$

Since

$$\sigma_n^c + \cdots + \sigma_{\overline{n}-1}^c = \sum_{n'=n}^{\overline{n}-1} \sigma_n^c \left(\frac{x_{n'}}{x_n}\right)^{1/p}$$

this gives

$$\sigma_n^c = \frac{\alpha_n^c - \alpha_{\overline{n}}^c}{\sum_{n'=n}^{\overline{n}-1}\left(\frac{x_{n'}}{x_n}\right)^{1/p}} = (\alpha_n^c - \alpha_{\overline{n}}^c)\left(\frac{x_n}{\widetilde{x}_{n:\overline{n}-1}}\right)^{1/p},$$

where $\widetilde{x}_{n:\overline{n}-1} \equiv \left(\sum_{n'=n}^{\overline{n}-1} {x_{n'}}^{1/p}\right)^p$. In this case, since $n = \underline{n}$, we obtain

$$\frac{x_{\underline{n}}}{\sigma_{\underline{n}}^{cp}} = \frac{\widetilde{x}_{\underline{n}:\overline{n}-1}}{(\alpha_{\underline{n}}^c - \alpha_{\overline{n}}^c)^p},$$

If $\overline{n} = J + 1$, then $\alpha_{\overline{n}} = 0$ and the same result is obtained.

Consider now n such that $\underline{n} < n \leq \overline{n} - 1$, for which $\tau_n = 0$,. Replacing n' by n in (A7) gives

$$\sigma_n^c = \sigma_{\underline{n}}^c \left(\frac{x_n}{x_{\underline{n}}} \right)^{1/p}.$$

This gives,

$$\frac{x_n}{\sigma_n^{cp}} = \frac{x_{\underline{n}}}{\sigma_{\underline{n}}^{cp}} = \frac{\widetilde{x}_{\underline{n}:\overline{n}-1}}{(\alpha_{\underline{n}}^c - \alpha_{\overline{n}}^c)^p}.$$

These expressions apply to the single completion case, since $\underline{n} = n$ and $\overline{n} = n + 1$.

The intuition behind these equations is that when a group of jobs complete simultaneously, the core allocations are distributed across these jobs in a job size-dependent way so that they complete together. This is similar to the optimal allocations across all jobs to minimize the makespan, except that in the current case only a subset of the jobs is included. □

A.2 Stage duration—Proof of Proposition 6.2

This section determines the duration of the stage τ_n. The proof of this requires an expression for λ_n in terms of job sizes. This is given by the following lemma.

Lemma A.7 *If $\tau_n > 0$ then*

$$\lambda_n = p \left(\frac{x_n}{\sigma_n^{cp}} - \frac{x_{\underline{\underline{n}}}}{\sigma_{\underline{\underline{n}}}^{cp}} \right) \frac{\alpha_n^c}{\psi_n}. \tag{A8}$$

Proof Recall that $\underline{n}$ is the largest integer less than n such that $\tau_{\underline{n}} > 0$. Assume initially that $\underline{\underline{n}} > 0$. Applying Lemma A.5 for n and $\underline{\underline{n}}$ gives, respectively,

$$p \frac{x_n}{\sigma_n^{cp}} = \sum_{j'=1, \tau'_j > 0}^{n} \frac{\lambda_{j'} \psi_{j'}}{\alpha_{j'}^c}$$

and

$$p \frac{x_{\underline{\underline{n}}}}{\sigma_{\underline{\underline{n}}}^{cp}} = \sum_{j'=1, \tau_{j'} > 0}^{\underline{\underline{n}}} \frac{\lambda_{j'} \psi_{j'}}{\alpha_{j'}^c}.$$

Subtracting the previous equation from the one prior to that gives

$$p\left(\frac{x_n}{\sigma_n^{cp}} - \frac{x_{\underline{\underline{n}}}}{\sigma_{\underline{\underline{n}}}^{cp}}\right) = \sum_{j'=1,\tau_{j'}>0}^{n} \frac{\lambda_{j'}\psi_{j'}}{\alpha_{j'}^{c}} - \sum_{j'=1,\tau_{j'}>0}^{\underline{\underline{n}}} \frac{\lambda_{j'}\psi_{j'}}{\alpha_{j'}^{c}} = \frac{\lambda_n\psi_n}{\alpha_n^{c}},$$

to obtain

$$\lambda_n = p\left(\frac{x_n}{\sigma_n^{cp}} - \frac{x_{\underline{\underline{n}}}}{\sigma_{\underline{\underline{n}}}^{cp}}\right)\frac{\alpha_n^{c}}{\psi_n}. \tag{A9}$$

The case for $\underline{\underline{n}} = 0$ gives the same result, where $x_{\underline{\underline{n}}} = 0$. □

Proof (Proposition 6.2)

To obtain τ_n, take any j such that $n \le j \le J$. Using Lemmas A.1 and A.2 gives

$$\tau_n = \frac{\lambda_n}{p}\frac{\theta_{j,n}}{\sigma_j}\frac{1}{s_{j,n}} = \frac{\lambda_n\psi_n^{1-p}}{p\alpha_n}. \tag{A10}$$

To complete the derivation of τ_n, applying Lemma A.7 to (A10) gives

$$\tau_n = p\left(\frac{x_n}{\sigma_n^{cp}} - \frac{x_{\underline{\underline{n}}}}{\sigma_{\underline{\underline{n}}}^{cp}}\right)\frac{\alpha_n^{c}}{\psi_n}\frac{\psi_n^{1-p}}{p\alpha_n} = \left(\frac{x_n}{\sigma_n^{cp}} - \frac{x_{\underline{\underline{n}}}}{\sigma_{\underline{\underline{n}}}^{cp}}\right)\frac{\alpha_n^{cp}}{\psi_n^{p}}. \tag{A11}$$

Substituting for σ_n in the stage duration equations in (A11) using Lemma (A.6) gives

$$\tau_n = \left(\frac{\widetilde{x}_{n:\overline{n}-1}}{(\alpha_n^{c} - \alpha_{\overline{n}}^{c})^{p}} - \frac{\widetilde{x}_{\underline{\underline{n}}:n-1}}{(\alpha_{\underline{\underline{n}}}^{c} - \alpha_n^{c})^{p}}\right)\frac{\alpha_n^{cp}}{\psi_n^{p}}.$$

□

A.3 Makespan—Proof of Proposition 6.3

Proof (Proposition 6.3)

Once the stage durations are known for the optimal allocation, the makespan and the total flow time can be computed. For makespan, substituting (34) into (35) gives (36) (repeated)

$$T = \sum_{n=1,\tau_n>0}^{J} \tau_n = \sum_{n=1,\tau_n>0}^{J}\left(\frac{x_n}{\sigma_n^{cp}} - \frac{x_{\underline{\underline{n}}}}{\sigma_{\underline{\underline{n}}}^{cp}}\right)\frac{\alpha_n^{cp}}{\psi_n^{p}}.$$

This can be rearranged to equal

$$T = \frac{x_M}{\sigma_M^{cp}}\alpha_M^{cp} + \sum_{n=1,\tau_n>0}^{M-1}\frac{x_n}{\sigma_n^{cp}}\left(\frac{\alpha_n^{cp}}{\psi_n^{p}} - \frac{\alpha_{\overline{n}}^{cp}}{\psi_{\overline{n}}^{p}}\right).$$

Since by Lemma A.6,

$$\frac{x_n}{\sigma_n^{cp}} = \frac{\widetilde{x}_{n:\overline{n}-1}}{\left(\alpha_n^c - \alpha_{\overline{n}}^c\right)^p},$$

(37) (repeated) follows:

$$T = \widetilde{x}_{M:J} + \sum_{n=1,\tau_n>0}^{M-1} \frac{\widetilde{x}_{n:\overline{n}-1}}{\left(\alpha_n^c - \alpha_{\overline{n}}^c\right)^p} \left(\frac{\alpha_n^{cp}}{\psi_n^p} - \frac{\alpha_{\overline{n}}^{cp}}{\psi_{\overline{n}}^p} \right).$$

□

A.4 Total flow time—Proof of Proposition 6.4

Proof (Proposition 6.4)

Expressions for total flow time can be obtained similarly to those for the makespan. Substituting (34) into (38) gives

$$\begin{aligned} S &= \sum_{n=1,\tau_n>0}^{J} (J-n+1)\tau_n \\ &= \sum_{n=1,\tau_n>0}^{J} (J-n+1)\left(\frac{x_n}{\sigma_n^{cp}} - \frac{x_{\underline{\underline{n}}}}{\sigma_{\underline{\underline{n}}}^{cp}}\right)\frac{\alpha_n^{cp}}{\psi_n^p}. \end{aligned}$$

Rearranging this and using Lemma A.6 gives

$$\begin{aligned} S &= (J-M+1)\frac{x_M}{\sigma_M^{cp}}\alpha_M^{cp} \\ &\quad + \sum_{n=1,\tau_n>0}^{M-1} \frac{x_n}{\sigma_n^{cp}} \left((J-n+1)\frac{\alpha_n^{cp}}{\psi_n^p} - (J-\overline{n}+1)\frac{\alpha_{\overline{n}}^{cp}}{\psi_{\overline{n}}^p} \right) \\ &= (J-M+1)\widetilde{x}_{M:J} \\ &\quad + \sum_{n=1,\tau_n>0}^{M-1} \frac{\widetilde{x}_{n:\overline{n}-1}}{\left(\alpha_n^c - \alpha_{\overline{n}}^c\right)^p} \left((J-n+1)\frac{\alpha_n^{cp}}{\psi_n^p} - (J-\overline{n}+1)\frac{\alpha_{\overline{n}}^{cp}}{\psi_{\overline{n}}^p} \right), \end{aligned}$$

where the final expression is (39). □

A.5 Viability of a potential solution

This section contains detailed derivations of the results given in Sect. 6.5 to determine the viability of a potential solution.

Lemma A.8 *For any completion pattern and $\psi_n = 1$, the makespan T is a decreasing function of μ ($M > 1$), and it is a constant for $M = 1$.*

Proof The coefficient of $x_{n:\overline{n}-1}$ in T in (37) is of the form

$$\frac{\alpha_n^{cp} - \alpha_{n+k}^{cp}}{(\alpha_n^c - \alpha_{n+k}^c)^p}, \tag{A12}$$

for some n and k. Lemma H.4 shows that this is decreasing in μ. For $M > 1$, since each term of T, except for the constant term $\widetilde{x}_{M:J}$, in (37) is decreasing and there is at least one non-constant term, T is a decreasing function of μ. If $M = 1$ only the constant term is present. □

Lemma A.9 *For any completion pattern and $\psi_n = 1$, the total flow time S is an increasing function of μ ($M > 1$), and is a constant for $M = 1$.*

Proof The coefficient of $x_{n:\overline{n}-1}$ in S in (39) is of the form

$$\frac{(J-n+1)\alpha_n^{cp} - (J-n+1-k)\alpha_{n+k}}{(\alpha_n^c - \alpha_{n+k})^p},$$

for some n and k. Lemma H.5 shows that this is increasing in μ. For $M > 1$, since each term of S, except for the constant term $(J-M+1)\widetilde{x}_{M:J}$, in (39) is increasing and there is at least one non-constant term, S is an increasing function of μ. If $M = 1$, only the constant term is present. □

A.5.1 τ_n-constraints

Lemma A.10 *For $M > 1$, τ_1 is a positive and increasing function of μ with $\tau_1 \to \infty$ ($\mu \to \infty$).*

Proof From (40), upon setting $j = \overline{1}$,

$$\tau_1 = \frac{\widetilde{x}_{1:j-1}}{(\alpha_1^c - \alpha_j^c)^p}\alpha_1^{cp} = \frac{\widetilde{x}_{1:j-1}}{(1 - \alpha_j^c/\alpha_1^c)^p}.$$

If $M > 1$, then $\alpha_j > 0$. It follows that τ_1 is increasing because $\alpha_j/\alpha_1 = (\alpha_1 - j)/\alpha_1 = 1 - j/\alpha_1$ is an increasing function of μ with limit 1 ($\mu \to \infty$). This implies that $\tau_1 \to \infty$. □

Lemma A.11 *For SJF ordering of jobs, $\tau_l \geq 0$ ($l = 2, \cdots, M-1$).*

Proof Only those indices l that have an active τ_n-constraint need to be considered. Given the completion epoch l, let n be the index of the next completion epoch ($n = J+1$ if it does not exist); let k be the index of the most recent completion epoch strictly before the index l ($k = 0$ if it does not exist). From (40), $\tau_l \geq 0$ ($n = 2, \cdots, M-1$) is equivalent to

$$\tau_l = \left(\frac{\widetilde{x}_{l:n-1}}{\sigma_{n:l-1}^{cp}} - \frac{\widetilde{x}_{k:l-1}}{\sigma_{k:l-1}^{cp}}\right)\alpha_l^{cp} \geq 0. \tag{A13}$$

The application of Lemma H.10(i) shows that the right-hand side is positive, and the result follows. □

Recall that $\mu_\triangle$ is defined as the value of μ for which $\tau_M = 0$. Properties of τ_M and $\mu_\triangle$ are given by the following lemma.

Lemma A.12 *(i) If $M = 1$, then there is no second last completion epoch, then $\underline{\underline{M}} = 0$, so that $\tau_M = \widetilde{x}_{1:J}$, and we set $\mu_\triangle = \infty$.*
(ii) If $M > 1$, then
(a) τ_M is a decreasing function of μ with $\tau_M \to -\infty$ ($\mu \to \infty$).
(b) For $\mu = 0$, $\tau_M(0) > 0$.
(c) $\mu_\triangle > 0$.
(d)

$$\mu_\triangle = \frac{M - \underline{\underline{M}}}{\left(\frac{\widetilde{x}_{M:J}}{\widetilde{x}_{\underline{\underline{M}}:J}}\right)^{(1-p)/p} - 1} - (J - M + 1),$$

where $\underline{\underline{M}}$ is the index of the second last completion epoch.

Proof (i) The case $M = 1$ is straightforward.
(ii) Suppose $M > 1$. Using (40), upon setting $j = \underline{\underline{M}}$, gives

$$\tau_M = \widetilde{x}_{M:J} - \frac{\widetilde{x}_{j:M-1}\alpha_M^{cp}}{(\alpha_j^c - \alpha_M^c)^p} = \widetilde{x}_{M:J} - \frac{\widetilde{x}_{j:M-1}}{((\alpha_j/\alpha_M)^c - 1)^p}. \tag{A14}$$

(a) We have that $\alpha_j/\alpha_M = (\alpha_M + (M - j))/\alpha_M = 1 + (M - j)/\alpha_M$ is a decreasing function of μ. Therefore, τ_M is a decreasing function of μ. Since $\alpha_j/\alpha_M \to 1$ ($\mu \to \infty$), it follows that $\tau_M \to -\infty$.
(b) We can express τ_M as

$$\tau_M = \widetilde{x}_{j:M-1}\left(\frac{\widetilde{x}_{M:J}}{\widetilde{x}_{j:M-1}} - \frac{\alpha_M^{cp}}{\sigma_{j:M-1}^c p}\right).$$

From Lemma H.8, $\widetilde{x}_{M:J}/\widetilde{x}_{j:M-1} \geq ((J - M + 1)/(M - j))^p$, while from Lemma H.9(ii)(a) $\alpha_M^{cp}/\sigma_{j:M-1}^{cp} \leq (\alpha_M/(M - j))^p$. For $\mu = 0$, $\alpha_M = J - M + 1$, which proves the result.
(c) Part (a) shows that τ_M is a decreasing function of μ with $\tau_M \to -\infty$ ($\mu \to \infty$). However, from (ii), $\tau_M(0) > 0$ and thus it must be the case that $\mu_\triangle > 0$ since all the functions involved are continuous.
(d) Set $j = \underline{\underline{M}}$. From lemma (A.13), $\tau_M = 0$ implies that

$$Y \equiv \left(\frac{\widetilde{x}_{j:J}}{\widetilde{x}_{M:J}}\right)^{1/cp} = \frac{\alpha_j}{\alpha_M} = \frac{J - j + 1 + \mu}{J - M + 1 + \mu}.$$

Solving for μ gives

$$\mu = \frac{(J - j + 1) - Y(J - M + 1)}{Y - 1}.$$

Further rearrangement gives

$$\mu = \frac{M - j}{Y - 1} - (J - M + 1),$$

which gives the stated result.

□

Lemma A.13 *Suppose $M > 1$ and $\tau_M = 0$, then*

$$\frac{\widetilde{x}_{M:J}^{1/p}}{\alpha_M^c} = \frac{\widetilde{x}_{j:J}^{1/p}}{\alpha_j^c}, \tag{A15}$$

where index $j = \underline{\underline{M}}$ is the index of the second last completion epoch.

Proof From (42), $\tau_M = 0$ gives

$$\frac{\widetilde{x}_{M:J}}{\sigma_{M:J}^{cp}} = \frac{\widetilde{x}_{j:M-1}}{\sigma_{j:M-1}^{cp}}. \tag{A16}$$

Thus,

$$\frac{\widetilde{x}_{j:M-1}}{\widetilde{x}_{M:J}} = \frac{\sigma_{j:M-1}^{cp}}{\sigma_{M:J}^{cp}} = \frac{(\alpha_j^c - \alpha_M^c)^p}{\alpha_M^{cp}}.$$

Hence,

$$\frac{\widetilde{x}_{j:M-1}^{1/p}}{\widetilde{x}_{M:J}^{1/p}} = \frac{\alpha_j^c - \alpha_M^c}{\alpha_M^c}.$$

This gives

$$\frac{\widetilde{x}_{j:J}^{1/p} - \widetilde{x}_{M:J}^{1/p}}{\widetilde{x}_{M:J}^{1/p}} = \frac{\widetilde{x}_{j:J}^{1/p}}{\widetilde{x}_{M:J}^{1/p}} - 1 = \frac{\alpha_j^c}{\alpha_M^c} - 1,$$

which proves the lemma. □

Appendix B Change in Makespan and Total Flow Time

Proof (Lemma 7.1)

The makespan and total flow time equations for pattern i ($i = 1, 2$), for the case $\psi_j = 1$, are given, respectively, by

$$T^i = \sum_{j=1,\tau_j>0}^{J} \frac{\widetilde{x}^{\pi}_{j:\overline{j}-1}}{\sigma^{cp}_{j:\overline{n}-1}} \left(\alpha^{cp}_j - \alpha^{cp}_{\overline{j}}\right),$$

$$S^i = \sum_{j=1,\tau_j>0}^{J} \frac{\widetilde{x}^{\pi}_{j:\overline{n}-1}}{\sigma^{cp}_{j:\overline{n}-1}} \left((J-j+1)\alpha^{cp}_j - (J-\overline{j}+1)\alpha^{cp}_{\overline{j}}\right).$$

where $\sigma^{c}_{j:\overline{j}-1} = \alpha^{c}_j - \alpha^{c}_{\overline{j}}$. Since μ is the same for patterns 1 and 2, the values of α_j ($= J - j + 1 + \mu$) are the same in both patterns. The distinguishing factor between the patterns is the completion pattern. The terms in the sums that differ are those corresponding to the indices k and l; the other terms ($j < k$ and $n \le j$) are not affected. Thus,

$$T^2 - T^1 = (T^2_k + T^2_l) - T^1_k,$$
$$S^2 - S^1 = (S^2_k + S^2_l) - S^1_k.$$

(i) Using Lemma B.1(i) the change in makespan is

$$\begin{aligned} T^2 - T^1 &= (T^2_k + T^2_l) - T^1_k \\ &= (b_k u_k + b_l u_l) - (a_k u_k + a_l u_l) = (\boldsymbol{b} - \boldsymbol{a}) \cdot \boldsymbol{u}. \end{aligned}$$

Substituting for the values of a_k, a_l, u_k, u_l gives

$$T^2 - T^1 = \widetilde{x}_{k:l-1} \frac{\alpha^{cp}_k - \alpha^{cp}_l}{\sigma^{cp}_{k:l-1}} + \widetilde{x}_{l:n-1} \frac{\alpha^{cp}_l - \alpha^{cp}_n}{\sigma^{cp}_{l:n-1}} - \widetilde{x}_{k:n-1} \frac{\alpha^{cp}_k - \alpha^{cp}_n}{\sigma^{cp}_{k:n-1}}.$$

(ii) (a)

Using Lemma B.1(ii) the change in total flow is

$$\begin{aligned} S^2 - S^1 &= (S^2_k + S^2_l) - S^1_k \\ &= (b_k v_k + b_l v_l) - (a_k v_k + a_l v_l) = (\boldsymbol{b} - \boldsymbol{a}) \cdot \boldsymbol{v}. \end{aligned}$$

Substituting for the values of a_k, a_l, v_k, v_l gives

$$\begin{aligned} S^2 - S^1 = \widetilde{x}_{k:l-1} \frac{(J-k+1)\alpha^{cp}_k - (J-l+1)\alpha^{cp}_l}{\sigma^{cp}_{k:l-1}} \\ + \widetilde{x}_{l:n-1} \frac{(J-l+1)\alpha^{cp}_l - (J-n+1)\alpha^{cp}_n}{\sigma^{cp}_{l:n-1}} \end{aligned}$$

$$-\widetilde{x}_{k:n-1}\frac{(J-k+1)\alpha_k^{cp}-(J-n+1)\alpha_n^{cp}}{\sigma_{k:n-1}^{cp}}.$$

(ii) (b) Using $J-k+1=\alpha_k-\mu$, and similarly for $J-l+1$ and $J-n+1$, the following decomposition can be derived.

$$\begin{aligned}S^2-S^1 &= \widetilde{x}_{k:l-1}\frac{\alpha_k^c-\alpha_l^c}{\sigma_{k:l-1}^{cp}}+\widetilde{x}_{l:n-1}\frac{\alpha_l^c-\alpha_n^c}{\sigma_{l:n-1}^{cp}}-\widetilde{x}_{k:n-1}\frac{\alpha_k^c-\alpha_n^c}{\sigma_{k:n-1}^{cp}}\\ &\quad-\mu\left(\widetilde{x}_{k:l-1}\frac{\alpha_k^{cp}-\alpha_l^{cp}}{\sigma_{k:l-1}^{cp}}+\widetilde{x}_{l:n-1}\frac{\alpha_l^{cp}-\alpha_n^{cp}}{\sigma_{l:n-1}^{cp}}-\widetilde{x}_{k:n-1}\frac{\alpha_k^{cp}-\alpha_n^{cp}}{\sigma_{k:n-1}^{cp}}\right)\\ &= (\widehat{S}^2-\widehat{S}^1)-\mu(T^2-T^1),\end{aligned}$$

where

$$\begin{aligned}\widehat{S}^1 &= \widetilde{x}_{k:n-1}\frac{\alpha_k^c-\alpha_n^c}{\sigma_{k:n-1}^{cp}}=\widetilde{x}_{k:n-1}(\alpha_k^c-\alpha_n^c)^{1-p},\\ \widehat{S}^2 &= \widetilde{x}_{k:l-1}\frac{\alpha_k^c-\alpha_l^c}{\sigma_{k:l-1}^{cp}}+\widetilde{x}_{l:n-1}\frac{\alpha_l^c-\alpha_n^c}{\sigma_{l:n-1}^{cp}}\\ &= \widetilde{x}_{k:l-1}(\alpha_k^c-\alpha_l^c)^{1-p}+\widetilde{x}_{l:n-1}(\alpha_l^c-\alpha_n^c)^{1-p}.\end{aligned}$$

□

Lemma B.1 *(i)*

$$\begin{aligned}T_k^1 &= \boldsymbol{a}\cdot\boldsymbol{u}=a_ku_k+a_lu_l\\ T_k^2+T_l^2 &= \boldsymbol{b}\cdot\boldsymbol{u}=b_ku_k+b_lu_l.\end{aligned}$$

(ii)

$$\begin{aligned}S_k^1 &= \boldsymbol{a}\cdot\boldsymbol{v}=a_kv_k+a_lv_l\\ S_k^2+S_l^2 &= \boldsymbol{b}\cdot\boldsymbol{v}=b_kv_k+b_lv_l.\end{aligned}$$

Proof We compute values before and after the split transformation. Recall that before the split transformation, the completion epochs that can contribute to a change in the makespan and the total flow time are at the indices k and n, and the split transformation introduces a new completion epoch at the index l: $k<l<n$.

Before the split transformation $\overline{k}=n$ and so the makespan term for index k in T^1 is

$$T_k^1=\frac{\widetilde{x}_{k:n-1}}{\sigma_{k:n-1}^{cp}}\left(\alpha_k^{cp}-\alpha_n^{cp}\right)=\frac{\widetilde{x}_{k:n-1}}{\sigma_{k:n-1}^{cp}}\left(\alpha_k^{cp}-\alpha_l^{cp}\right)+\frac{\widetilde{x}_{k:n-1}}{\sigma_{k:n-1}^{cp}}\left(\alpha_l^{cp}-\alpha_n^{cp}\right).$$

This can be expressed as $T_k^1=a_ku_k+a_lu_l$, where

$$a_k=\frac{\widetilde{x}_{k:n-1}}{\sigma_{k:n-1}^{cp}},\quad a_l=\frac{\widetilde{x}_{k:n-1}}{\sigma_{k:n-1}^{cp}},\qquad \text{and, } u_k=\alpha_k^{cp}-\alpha_l^{cp},\quad u_l=\alpha_l^{cp}-\alpha_n^{cp}.$$

After the split transformation, $\bar{k} = l$, $\bar{l} = n$, and so the sum of the makespan components due to indices k and l in T^2 is given by

$$T_k^2 + T_l^2 = \frac{\widetilde{x}_{k:l-1}}{\sigma_{k:l-1}^{cp}} \left(\alpha_k^{cp} - \alpha_l^{cp}\right) + \frac{\widetilde{x}_{l:n-1}}{\sigma_{l:n-1}^{cp}} \left(\alpha_l^{cp} - \alpha_n^{cp}\right).$$

This can be expressed as $T_k^2 + T_l^1 = b_k u_k + b_l u_l$, where

$$b_k = \frac{\widetilde{x}_{k:l-1}}{\sigma_{k:l-1}^{cp}}, \qquad b_l = \frac{\widetilde{x}_{l:n-1}}{\sigma_{l:n-1}^{cp}}.$$

The total flow time component in S^1 before the split transformation, due to index k, is given by

$$\begin{aligned} S_k^1 &= \frac{\widetilde{x}_{k:n-1}}{\sigma_{k:n-1}^{cp}} \left((J-k+1)\alpha_k^{cp} - (J-n+1)\alpha_n^{cp}\right) \\ &= \frac{\widetilde{x}_{k:n-1}}{\sigma_{k:n-1}^{cp}} \left((J-k+1)\alpha_k^{cp} - (J-l+1)\alpha_l^{cp}\right) \\ &\quad + \frac{\widetilde{x}_{k:n-1}}{\sigma_{k:n-1}^{cp}} \left((J-l+1)\alpha_l^{cp} - (J-n+1)\alpha_n^{cp}\right). \end{aligned}$$

This can be expressed as $S_k^1 = a_k v_k + a_l v_l$, where

$$\begin{aligned} v_k &= (J-k+1)\alpha_k^{cp} - (J-l+1)\alpha_l^{cp}, \\ v_l &= (J-l+1)\alpha_l^{cp} - (J-n+1)\alpha_n^{cp}. \end{aligned}$$

The sum of the total flow time components in S^2 after the split transformation, due to indices k and l, is given by

$$\begin{aligned} S_k^2 + S_l^2 &= \frac{\widetilde{x}_{k:l-1}}{\sigma_{k:l-1}^{cp}} \left((J-k+1)\alpha_k^{cp} - (J-l+1)\alpha_l^{cp}\right) \\ &\quad + \frac{\widetilde{x}_{l:n-1}}{\sigma_{l:n-1}^{cp}} \left((J-l+1)\alpha_l^{cp} - (J-n+1)\alpha_n^{cp}\right). \end{aligned}$$

This can be expressed as $S_k^1 + S_l^1 = b_k v_k + b_l v_l$. □

Appendix C Optimality of scenarios

The proofs given in this section are for the more general case where the split can occur at any index greater than the index of the last completion epoch and not just the next index (See Remark 4). In the remainder of this section, pattern 1 is the pattern before the end-split transformation and pattern 2 is the pattern after the end-split transformation.

In the notation of Sect. 7.2, $k = M^1$ is the index of the final completion index in pattern 1. Let the split occur at index l, for $k < l \leq J$. In this case, the index of the final completion index in pattern 2 is $M^2 = l$.

C.1 Intervals

Proposition C.1 *For an end-split transformation:*

$$(i) \quad T^2(\mu) - T^1(\mu) \quad \begin{cases} > 0, \ \mu < \mu_{\triangle}^2 \\ = 0, \ \mu = \mu_{\triangle}^2 \\ < 0, \ \mu > \mu_{\triangle}^2 \end{cases}.$$

Proof For an end-split transformation, $n = J+1, \alpha_{J+1} = 0$, and $\sigma_{k:J}^{cp} = \alpha_k^{cp}$. Inserting these values into (55) gives

$$T^2 - T^1 = \widetilde{x}_{k:l-1} \frac{\alpha_k^{cp} - \alpha_l^{cp}}{\sigma_{k:l-1}^{cp}} + \widetilde{x}_{l:J} - \widetilde{x}_{k:J} = T_B + \widetilde{x}_{l:J} - \widetilde{x}_{k:J}. \tag{C17}$$

The only terms involving μ are those associated with $\widetilde{x}_{k:l-1}$.

We first show that if $\mu = \mu_{\triangle}^2$ then $T^2 - T^1 = 0$. For pattern 2, the final completion index is $M = l$ and the second-last completion index is $\underline{\underline{M}} = k$. Setting $\tau_M = 0$ in (A14) with M replaced by l and j replaced by k gives

$$0 = \widetilde{x}_{l:J} - \frac{\widetilde{x}_{k:l-1}}{\sigma_{k:l-1}^{cp}} \alpha_l^{cp},$$

so that

$$\alpha_l^{cp} = \sigma_{k:l-1}^{cp} \frac{\widetilde{x}_{l:J}}{\widetilde{x}_{k:l-1}}. \tag{C18}$$

With this value of α_l^{cp},

$$T_B = \widetilde{x}_{k:l-1} \frac{\alpha_k^{cp} - \sigma_{k:l-1}^{cp} \frac{\widetilde{x}_{l:J}}{\widetilde{x}_{k:l-1}}}{\sigma_{k:l-1}^{cp}} = \widetilde{x}_{k:l-1} \frac{\alpha_k^{cp}}{\sigma_{k:l-1}^{cp}} - \widetilde{x}_{l:J},$$

and so

$$\begin{aligned} T_B + \widetilde{x}_{l:J} &= \widetilde{x}_{k:l-1} \frac{\alpha_k^{cp}}{\sigma_{k:l-1}^{cp}} - \widetilde{x}_{l:J} + \widetilde{x}_{l:J} \\ &= \widetilde{x}_{k:l-1} \frac{\alpha_k^{cp}}{\sigma_{k:l-1}^{cp}} = \widetilde{x}_{l:J} \frac{\alpha_k^{cp}}{\alpha_l^{cp}} = \widetilde{x}_{k:J}, \end{aligned}$$

where (C18) has been used to obtain the second last equality, and Lemma A.13 has been used to obtain the last equality. This completes the proof for $\mu = \mu^2_\triangle$.

Using Lemma H.4, $(\alpha^{cp}_k - \alpha^{cp}_l)/\sigma^{cp}_{k:l-1}$ is a decreasing function of μ. Applied to (C17), this gives $T^2 - T^1 > 0$, for $\mu < \mu^2_\triangle$, and $T^2 - T^1 < 0$, for $\mu^2_\triangle < \mu$. □

Lemma C.1 *For an end-split transformation,* $\mu^2_\triangle < \mu^1_\triangle$.

Proof If pattern 1 equals scenario 1, then $\mu^1_\triangle = \infty$. Since, in this case, pattern 2 has two completion epochs, $\mu^2_\triangle < \infty$.

Suppose now that pattern 1 does not equal scenario 1, in which case it has at least two completion epochs. Let j be the index of the second last completion epoch for pattern 1. Then,

$$\mu^1_\triangle = \frac{M^1 - j}{\left(\frac{\widetilde{x}_{j:J}}{\widetilde{x}_{M^1:J}}\right)^{(1-p)/p} - 1} - (J - M^1 + 1).$$

Since M^1 is the index of the second last completion epoch in pattern 2,

$$\mu^2_\triangle = \frac{M^2 - M^1}{\left(\frac{\widetilde{x}_{M^1:J}}{\widetilde{x}_{M^2:J}}\right)^{(1-p)/p} - 1} - (J - M^2 + 1).$$

For ease of notation define

$$a = \widetilde{x}^{1/p}_{j:J}, \qquad b = \widetilde{x}^{1/p}_{j:M^1}, \qquad c = \widetilde{x}^{1/p}_{M^1:J}, \qquad d = \widetilde{x}^{1/p}_{M^1:M^2}, \qquad e = \widetilde{x}^{1/p}_{M^2:J}.$$

$\mu^2_\triangle < \mu^1_\triangle$ requires

$$\frac{M^2 - M^1}{\left(\frac{c}{e}\right)^{1-p} - 1} - (J - M^2 + 1) \le \frac{M^1 - j}{\left(\frac{a}{c}\right)^{1-p} - 1} - (J - M^1 + 1).$$

Rearranging this gives

$$\frac{a^{1-p} - c^{1-p}}{c^{1-p} - e^{1-p}} \le \frac{M^1 - j}{M^2 - M^1}. \tag{C19}$$

Using $a = b + c$ and $c = d + e$, the left-hand side becomes

$$\frac{(b+c)^{1-p} - c^{1-p}}{(d+e)^{1-p} - e^{1-p}} = \frac{(b+c)^{1-p} - c^{1-p}}{b} \times \frac{d}{(d+e)^{1-p} - e^{1-p}} \times \frac{b}{d}.$$

Applying the mean value theorem, this equals

$$\frac{(1-p)(c+\phi b)^{-p}}{(1-p)(e+\psi d)^{-p}} \times \frac{b}{d},$$

where $0 \le \phi, \psi, \le 1$. Since $c + \phi b \ge c$ and $e + \psi d \le e + d = c$,

$$\frac{c + \phi b}{e + \psi d} \ge 1.$$

However, $-p < 0$ and so the left-hand side of (C19) is less than b/d. Thus, if we show that

$$\frac{b}{d} = \frac{\widetilde{x}^{1/p}_{j:M^1}}{\widetilde{x}^{1/p}_{M^1:M^2}} \le \frac{M^1 - j}{M^2 - M^1},$$

then the result is true. This inequality is true from Lemma H.8(i), which proves the lemma. □

Lemma C.2 *Suppose patterns 2 and 2′ are each derived from pattern 1 by means of an end-split transformation. Let the index of the final completion epoch for pattern 2 be M^2 and the index of the final completion epoch for pattern 2′ be $M^{2'}$. If $M^{2'} < M^2$, then $\mu^2_\triangle < \mu^{2'}_\triangle$.*

Proof Since M^1 is the index of the second last completion epoch for patterns 2 and 2′,

$$\mu^2_\triangle = \frac{M^2 - M^1}{\left(\frac{\widetilde{x}_{M^1:J}}{\widetilde{x}_{M^2:J}}\right)^{(1-p)/p} - 1} - (J - M^2 + 1),$$
$$\mu^{2'}_\triangle = \frac{M^{2'} - M^1}{\left(\frac{\widetilde{x}_{M^1:J}}{\widetilde{x}_{M^{2'}:J}}\right)^{(1-p)/p} - 1} - (J - M^{2'} + 1).$$

For ease of notation define

$$c = \widetilde{x}^{1/p}_{M^1:J}, \quad d = \widetilde{x}^{1/p}_{M^1:M^{2'}}, \quad e = \widetilde{x}^{1/p}_{M^2:J}, \quad e' = \widetilde{x}^{1/p}_{M^{2'}:J}, \quad f = \widetilde{x}^{1/p}_{M^{2'}:M^2}.$$

$\mu^2_\triangle < \mu^{2'}_\triangle$ requires

$$\frac{M^2 - M^1}{\left(\frac{c}{e}\right)^{1-p} - 1} - (J - M^2 + 1) \le \frac{M^{2'} - M^1}{\left(\frac{c}{e'}\right)^{1-p} - 1} - (J - M^{2'} + 1).$$

Rearranging this gives

$$\frac{M^2 - M^1}{M^{2'} - M^1} = \frac{M^2 - M^1}{M^{2'} - M^1} + 1 \le \frac{c^{1-p} - e^{1-p}}{c^{1-p} - e'^{1-p}}$$
$$= 1 + \frac{e'^{1-p} - e^{1-p}}{c^{1-p} - e'^{1-p}}.$$

Thus, the following is required to demonstrated:

$$\frac{M^2 - M^{2'}}{M^{2'} - M^1} \leq \frac{e'^{1-p} - e^{1-p}}{c^{1-p} - e'^{1-p}}. \tag{C20}$$

Using $c = e' + d$ and $e' = e + f$, the right-hand side becomes

$$\frac{(f+e)^{1-p} - e^{1-p}}{(d+e')^{1-p} - e'^{1-p}} = \frac{(f+e)^{1-p} - e^{1-p}}{f} \times \frac{d}{(d+e')^{1-p} - e'^{1-p}} \times \frac{f}{d}.$$

Applying the mean value theorem, this equals

$$\frac{(1-p)(e+\phi f)^{-p}}{(1-p)(e'+\psi d)^{-p}} \times \frac{f}{d},$$

where $0 \leq \phi, \psi, \leq 1$. Since $e + \phi f \leq e + f = e'$ and $e' + \psi d \geq e'$,

$$\frac{e+\phi f}{e'+\psi d} \leq 1.$$

However, $-p < 0$ and so the right-hand side of (C20) is greater than f/d. Thus, if we show

$$\frac{M^2 - M^{2'}}{M^{2'} - M^1} < \frac{f}{d} = \frac{\widetilde{x}^{1/p}_{M^{2'}:M^2}}{\widetilde{x}^{1/p}_{M^1:M^{2'}}}$$

then the result is true. The inequality is true from Lemma H.8(i), which proves the lemma. □

C.2 Total flow time—single value of μ

Proposition C.2 *For an end-split transformation:*

$$S^2(\mu) - S^1(\mu) \begin{cases} < 0, \ \mu < \mu^2_{\triangle} \\ = 0, \ \mu = \mu^2_{\triangle} \\ > 0, \ \mu > \mu^2_{\triangle} \end{cases}.$$

Proof Inserting $n = J + 1$, $\alpha_{J+1} = 0$, and $\sigma^{cp}_{k:J} = \alpha^{cp}_k$ into (57) gives:

$$\begin{aligned} S^2 - S^1 = \frac{\widetilde{x}_{k:l-1}}{\sigma^{cp}_{k:l-1}} \left[(J-k+1)\alpha^{cp}_k - (J-l+1)\alpha^{cp}_l \right] \\ -(J-k+1)\widetilde{x}_{k:J} + (J-l+1)\widetilde{x}_{l:J}. \end{aligned} \tag{C21}$$

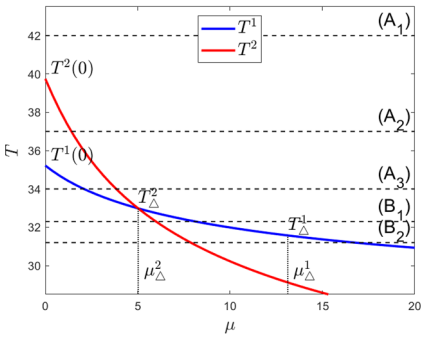

Fig. 6 End-split transformation for different cases for t_0

From (58) $S^2 - S^1 = (\widehat{S}^2 - \widehat{S}^1) - \mu(T^2 - T^1)$. We first show that if $\mu = \mu^2_\triangle$, then $S^2 - S^1 = 0$. From (i), $T^2 - T^1 = 0$; therefore, what is required to be proven is $\widehat{S}^2 = \widehat{S}^1$. From (59) and (60), and noting $n = J + 1$, so that $\alpha_n = 0$, $(\alpha^c_k)^{1-p} = \alpha_k$ and similarly for α_l,

$$\begin{aligned}\widehat{S}_1 &= \widetilde{x}_{k:J}\alpha_k\\ \widehat{S}_2 &= \widetilde{x}_{k:l-1}(\alpha^c_k - \alpha^c_l)^{1-p} + \widetilde{x}_{l:J}\alpha_l.\end{aligned}$$

From Lemma A.13,

$$\alpha_l = \alpha_k \left(\frac{\widetilde{x}_{l:J}}{\widetilde{x}_{k:J}}\right)^{1/cp}.$$

This and (C18) give

$$\alpha^c_k - \alpha^c_l = \alpha^c_l \left(\frac{\widetilde{x}_{l:J}}{\widetilde{x}_{k:l-1}}\right)^{1/p} = \alpha^c_k \left(\frac{\widetilde{x}_{k:l-1}}{\widetilde{x}_{k:J}}\right)^{1/p}.$$

This gives

$$\begin{aligned}\widehat{S}^2 &= \widetilde{x}_{k:l-1}\alpha_k \left(\frac{\widetilde{x}_{k:l-1}}{\widetilde{x}_{k:J}}\right)^{(1-p)/p} + \widetilde{x}_{l:J}\alpha_k \left(\frac{\widetilde{x}_{l:J}}{\widetilde{x}_{k:J}}\right)^{(1-p)/p}\\ &= \frac{\alpha_k}{\widetilde{x}^{(1-p)/p}_{k:J}}\left(\widetilde{x}_{k:l-1} + \widetilde{x}^{1/p}_{l:J}\right) = \alpha_k \widetilde{x}_{kJ} = \widehat{S}^1.\end{aligned}$$

This completes the proof for $\mu = \mu^2_\triangle$. The cases of $\mu > \mu^2_\triangle$ and $\mu > \mu^2_\triangle$ follow from (C21). Lemma H.5 shows that the first term on the right-hand side of (C21), $((J - k + 1)\alpha^{cp}_k - (J - l + 1)\alpha^{cp}_l)/\sigma^{cp}_{k:l-1}$, is an increasing function of μ with limit infinity. This is sufficient to prove the result. □

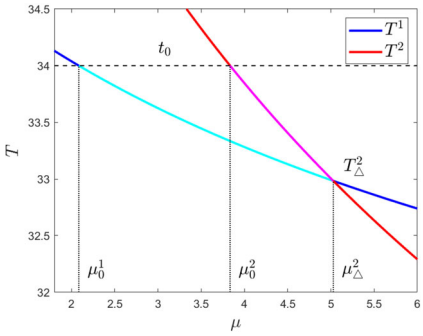

Fig. 7 End-split transformation: T

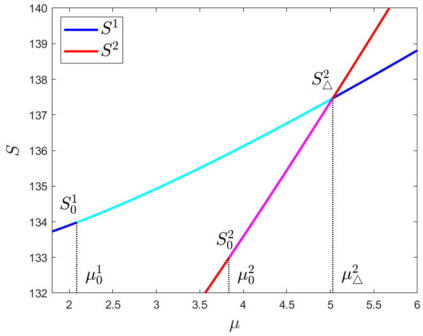

Fig. 8 End-split transformation: S

C.3 Total flow time—t_0-constraint applied

Lemma C.3 *For an end-split transformation:*

(i) If $T^2_\triangle \le t_0$, *then* $S^2_0 \le S^1_0$.
(ii) If $T^1(\infty) < t_0 < T^2_\triangle$, *then* $S^2_0 > S^1_0$.

Proof (i) When applied to scenarios, this corresponds to case (A) in Table 4. Case (A) is divided into three subcases: (A_1), (A_2), and (A_3). (See Fig. 6).

Case (A_1) $(T^2(0) < t_0)$. In this case $\mu^1_0 = \mu^2_0 = 0$, since $\mu^2_0 < \mu^2_\triangle$, Proposition C.2 gives $S^2_0 < S^1_0$.

Case (A_3) is considered before case (A_2).

Case (A_3) $(T^2_\triangle \le t_0 < T^1(0))$. This case corresponds to $0 < \mu^1_0 < \mu^2_0 \le \mu^2_\triangle$. The sensitivity results for the Lagrange multipliers give, for a general total flow time S, the change in total flow time for a given change in the t_0 constraint. In particular, if the constraint increases from t_1 to t_2, the change in total flow time is

$$\Delta S = -\int_{t_1}^{t_2} \mu(t)\mathrm{d}t, \tag{C22}$$

where the function $\mu(t) = T^{-1}(t)$ is the inverse mapping from the t to the corresponding Lagrange multiplier, μ. Define $S^2_\triangle = S^2(\mu^2_\triangle)$. Note that $S^2_\triangle = S^1(\mu^2_\triangle)$ and

$T^2_\triangle = T^1(\mu^2_\triangle)$. Setting $t_1 = T^2_\triangle$ and $t_2 = t_0$, gives

$$S^1_0 - S^2_\triangle = -\int_{T^2_\triangle}^{t_0} \mu_1(t)\mathrm{d}t,$$

and

$$S^2_0 - S^2_\triangle = -\int_{T^2_\triangle}^{t_0} \mu_2(t)\mathrm{d}t.$$

(See Figs. 7 and 8.) This gives

$$S^2_0 - S^1_0 = (S^2_0 - S^2_\triangle) - (S^1_0 - S^2_\triangle) = -\int_{T^2_\triangle}^{t_0} (\mu_2(t) - \mu_1(t))\mathrm{d}t.$$

Over the range of integration, $T^2(\mu) > T^1(\mu)$ giving $\mu_1(t) < \mu_2(t)$, and so the integrand is positive. If $t_0 = T^2_\triangle$, then $S^2_0 - S^1_0 = 0$; otherwise, $S^2_0 - S^1_0 < 0$.

Case (A_2) ($T^1(0) \le t_0 < T^2_0$). This case corresponds to $0 = \mu^1_0 < \mu^2_0 < \mu^2_\triangle$. In this case,

$$S^1(0) - S^2_\triangle = -\int_{T^2_\triangle}^{T^1(0)} \mu_1(t)\mathrm{d}t,$$

and

$$S^2_0 - S^1(\mu^2_\triangle) = -\int_{T^2_\triangle}^{t_0} \mu_2(t)\mathrm{d}t.$$

Hence,

$$\begin{aligned} S^2_0 - S^1(0) &= (S^2_0 - S^2_\triangle) - (S^1(0) - S^2_\triangle) \\ &= -\left[\int_{T^2_\triangle}^{t_0} \mu_2(t)\mathrm{d}t - \int_{T^2_\triangle}^{T^1(0)} \mu_1(t)\mathrm{d}t\right] \\ &= -\int_{T^2_\triangle}^{T^1(0)} (\mu_2(t) - \mu_1(t))\mathrm{d}t - \int_{T^1(0)}^{t_0} \mu_2(t)\mathrm{d}t < 0. \end{aligned}$$

(ii) This combines the cases (B_1) and (B_2) in Table 4 ($T^1(\infty) < t_0 < T^2_\triangle$), for which $\mu^2_\triangle < \mu^2_0 < \mu^1_0$. This is the same as in Case (A_3), except that $\mu^2(t) < \mu^1(t)$, which gives $S^2_0 - S^1_0 > 0$. □

Appendix D Optimization over all patterns

In this section, pattern 1 refers to the pattern before the internal-split transformation, and pattern 2 refers to the pattern before the internal-split transformation. In the notation of Sect. 7.2, consecutive completion epochs occur at the end of stages k and n with $k < n \leq J$ and the split occurs at index l ($k < l < n$).

D.1 Intervals

Proof (Lemma 9.2)

(i) In this case, $\tau^2_{M_2}(\mu) = \tau^1_{M_1}(\mu)$, since none of the terms in τ_M is affected by the internal-split transformation. Therefore, $\mu^2_\triangle = \mu^1_\triangle$.

(ii) We first show that $\tau^2_{M_2}(\mu) < \tau^1_{M_1}(\mu)$. From (A11),

$$\tau^1_{M_1} = \widetilde{x}_{M:J} - \frac{\widetilde{x}_{k:M-1}}{\sigma^{cp}_{k:M-1}}\alpha^{cp}_M, \qquad \tau^2_{M_2} = \widetilde{x}_{M:J} - \frac{\widetilde{x}_{l:M-1}}{\sigma^{cp}_{l:M-1}}\alpha^{cp}_M.$$

Thus, we have $\tau^2_{M_2} < \tau^1_{M_1}$ if

$$\widetilde{x}_{M:J} - \frac{\widetilde{x}_{l:M-1}}{\sigma^{cp}_{l:M-1}}\alpha^{cp}_M \leq \widetilde{x}_{M:J} - \frac{\widetilde{x}_{k:M-1}}{\sigma^{cp}_{k:M-1}}\alpha^{cp}_M;$$

that is, if

$$\frac{\widetilde{x}_{k:M-1}}{\sigma^{cp}_{k:M-1}} < \frac{\widetilde{x}_{l:M-1}}{\sigma^{cp}_{l:M-1}}.$$

This follows from Lemma H.10(iii). Since $\tau^2_{M_2}(\mu^2_\triangle) = 0$, $\tau^1_{M_1}(\mu^2_\triangle) > 0$. This implies, since $\tau^1_{M_1}$ is a continuous and decreasing function, that $\mu^2_\triangle < \mu^1_\triangle$.

□

Proof (Lemma 9.3.) (i) Numerical studies show that both possibilities can occur. This can be explained as follows. Lemma 9.2 shows for $n < M$ that $\mu^1_\triangle = \mu^2_\triangle = \mu_\triangle$. As explained in Sect. 9.4, makespan can increase or decrease for a given value of μ for an internal-split transformation. In this case, this gives the two possibilities $T^1(\mu_\triangle) < T^2(\mu_\triangle)$ and $T^2(\mu_\triangle) < T^1(\mu_\triangle)$.

(ii) If $n < M$, then Lemma 9.2 gives $\mu^2_\triangle < \mu^1_\triangle$. We proceed as follows.

Let pattern $1'$ be obtained from pattern 1 by having all the completions that occur at index n moved to index k. That is, the "1" at index n in the completion pattern becomes a "0". Pattern $1'$ can be transformed into pattern 1 using an end-split transformation at index n. Let pattern $2'$ be obtained from pattern 2 by having all completions at index n be moved to index l. That is, the "1" at index n in the completion pattern becomes a "0". Pattern $1'$ can be transformed into pattern $2'$ by means of an end-split transformation at index l and pattern $2'$ can be transformed into pattern 2 by means of an end-split

transformation at index n. As an example, for $J = 8$, if pattern 1 = 11000010 and pattern 2 = 11001010 then pattern $1'$ = 11000000 and pattern $2'$ = 11001000. In this case, $k = 2, l = 5$ and $n = 7$.

Using Remark 4, the results for scenarios, in particular Lemma 8.3, can also be applied to any sequence of patterns obtained by using end-split transformations. The proof now proceeds through the following steps.

(i) To begin, $T^1_\triangle = T^1(\mu^1_\triangle)$.

(ii) Since pattern 1 is obtained from pattern $1'$ by means of an end-split transformation, Proposition 8.1 gives

$$T^1(\mu^1_\triangle) = T^{1'}(\mu^1_\triangle).$$

(iii) Since patterns 1 and $2'$ are derived from pattern $1'$ by means of an end-split transformation with $M^{2'} = l < M^1 = n$, Lemma C.2 gives $\mu^1_\triangle < \mu^{2'}_\triangle$.

(iv) Since pattern $2'$ is obtained from pattern $1'$ by means of an end-split transformation, and $\mu^1_\triangle < \mu^{2'}_\triangle$, Proposition 8.1 gives

$$T^{2'}(\mu^1_\triangle) > T^{1'}(\mu^1_\triangle).$$

(v) For an internal-split transformation with $n = M$, Lemma 9.2 gives $\mu^2_\triangle < \mu^1_\triangle$. Since $T^{2'}$ is decreasing,

$$T^{2'}(\mu^2_\triangle) > T^{2'}(\mu^1_\triangle).$$

(vi) Since pattern 2 is obtained from pattern $2'$ by means of an end-split transformation

$$T^2(\mu^2_\triangle) = T^{2'}(\mu^2_\triangle).$$

(vii) To complete the proof, $T^2_\triangle = T^2(\mu^2_\triangle)$. □

D.2 Total flow time—single value of μ

Proof (Proof of Proposition 9.2)

The two possibilities for change in makespan are analyzed separately. Lemma D.1 proves the result for the case where makespan decreases. Lemma D.2 proves the result for the case where makespan increases. □

Makespan decreases

This section considers the case where the makespan reduces, so that $T^2 \leq T^1$.

Lemma D.1 *For an internal-split transformation, if $T^2 \leq T^1$, then $S^2 \leq S^1$.*

Proof Using (56) and (54), we can write

$$\begin{aligned} S^2 - S^1 &= (\boldsymbol{b} - \boldsymbol{a}) \cdot \boldsymbol{v} - (T^2 - T^1) + (T^2 - T^1) \\ &= (\boldsymbol{b} - \boldsymbol{a}) \cdot \boldsymbol{v} - (\boldsymbol{b} - \boldsymbol{a}) \cdot \boldsymbol{u} + (T^2 - T^2) \\ &= (\boldsymbol{b} - \boldsymbol{a}) \cdot (\boldsymbol{v} - \boldsymbol{u}) + (T^2 - T^1). \end{aligned}$$

Hence, to prove $S^2 - S^1 < 0$ it suffices to show that $(\boldsymbol{b} - \boldsymbol{a}) \cdot (\boldsymbol{v} - \boldsymbol{u}) < 0$. Expanding the preceding displayed expression, this requirement becomes

$$(\boldsymbol{b} - \boldsymbol{a}) \cdot (\boldsymbol{v} - \boldsymbol{u}) = (b_k - a_k)(v_k - u_k) + (b_l - a_l)(v_l - u_l) < 0,$$

which can be rewritten as

$$u_k(b_k - a_k)\left(\frac{v_k}{u_k} - 1\right) + u_l(b_l - a_l)\left(\frac{v_l}{u_l} - 1\right) < 0.$$

We show that $b_k - a_k < 0$ and $b_l - a_l > 0$. For the former, inserting the definitions of a_k, a_l, b_k, b_l gives, upon using Lemma H.10(ii),

$$b_k - a_k = \frac{\widetilde{x}_{k:l-1}}{\sigma^{cp}_{k:l-1}} - \frac{\widetilde{x}_{k:n-1}}{\sigma^{cp}_{k:n-1}} < 0.$$

It can be shown similarly that $b_l - a_l > 0$.

Consider now the terms that include u and v. From Lemma H.11, it follows that $v_l/u_l < v_k/u_k$ since

$$\begin{aligned} \frac{v_l}{u_l} &= \frac{(J - l + 1)\alpha^{cp}_l - (J - n + 1)\alpha^{cp}_n}{\alpha^{cp}_l - \alpha^{cp}_n} \\ &< \frac{(J - k + 1)\alpha^{cp}_k - (J - l + 1)\alpha^{cp}_l}{\alpha^{cp}_k - \alpha^{cp}_l} = \frac{v_k}{u_k}. \end{aligned}$$

Also, $1 < v_l/u_l$ since

$$\begin{aligned} \frac{v_l}{u_l} &= \frac{(J - l + 1)\alpha^{cp}_l - (J - n + 1)\alpha^{cp}_n}{\alpha^{cp}_l - \alpha^{cp}_n} \\ &= \frac{(J - l + 1)\alpha^{cp}_l - (J - l + 1)\alpha^{cp}_n}{\alpha^{cp}_l - \alpha^{cp}_n} + \frac{(J - l + 1)\alpha^{cp}_n - (J - n + 1)\alpha^{cp}_n}{\alpha^{cp}_l - \alpha^{cp}_n} \\ &= (J - l + 1) + \frac{(n - l)\alpha^{cp}_n}{\alpha^{cp}_l - \alpha^{cp}_n} > 1. \end{aligned}$$

This gives

$$u_k(b_k - a_k)\left(\frac{v_k}{u_k} - 1\right) + u_l(b_l - a_l)\left(\frac{v_l}{u_l} - 1\right)$$

$$< u_k(b_k - a_k)\left(\frac{v_l}{u_l} - 1\right) + u_l(b_l - a_l)\left(\frac{v_l}{u_l} - 1\right)$$
$$= \left(\frac{v_l}{u_l} - 1\right)\big(u_k(b_k - a_k) + u_l(b_l - a_l)\big) = \left(\frac{v_l}{u_l} - 1\right)(T^2 - T^1) < 0.$$

The second line follows since the term containing $b_k - a_k$ is negative, and it has been reduced in magnitude. □

Makespan increases

This section examines the case for an internal-split transformation, when the makespan increases, that is, when $T^2 > T^1$.

Lemma D.2 *For an internal-split transformation, if $T^2 > T^1$, then $S^2 < S^1$.*

Proof In this case, the method of the previous section cannot be applied. Instead, the proof of $S^2 - S^1 < 0$ begins by decomposing $S^2 - S^1$ into two components, as described by (58):

$$S^2 - S^1 = (\widehat{S}^2 - \widehat{S}^1) - \mu(T^2 - T^1).$$

Lemma D.3 proves that $\widehat{S}^2 - \widehat{S}^1 < 0$. This, combined with the assumption that $T^2 - T^1 > 0$, together with $\mu \geq 0$, completes the proof. □

Lemma D.3 *For given μ, $\widehat{S}^2 < \widehat{S}^1$, that is,*

$$\widetilde{x}_{k:l-1}(\alpha_k^c - \alpha_l^c)^{1-p} + \widetilde{x}_{l:n-1}(\alpha_l^c - \alpha_n^c)^{1-p} < \widetilde{x}_{k:n-1}(\alpha_k^c - \alpha_n^c)^{1-p}. \quad \text{(D23)}$$

Proof (D23) is equivalent to

$$\widetilde{x}_{k:l-1}^{1/p}\frac{(\alpha_k^c - \alpha_l^c)^{1-p}}{\widetilde{x}_{k:l-1}^{1/p-1}} + \widetilde{x}_{l:n-1}^{1/p}\frac{(\alpha_l^c - \alpha_n^c)^{1-p}}{\widetilde{x}_{l:n-1}^{1/p-1}} < \widetilde{x}_{k:n-1}^{1/p}\frac{(\alpha_k^c - \alpha_n^c)^{1-p}}{\widetilde{x}_{k:n-1}^{1/p-1}}.$$

This can be expressed as

$$p_1\left(\frac{(\alpha_k^c - \alpha_l^c)}{\widetilde{x}_{k:l-1}^{1/p}}\right)^{1-p} + p_2\left(\frac{(\alpha_l^c - \alpha_n^c)}{\widetilde{x}_{l:n-1}^{1/p}}\right)^{1-p} < \left(\frac{(\alpha_k^c - \alpha_n^c)}{\widetilde{x}_{k:n-1}^{1/p}}\right)^{1-p},$$

where

$$p_1 = \frac{\widetilde{x}_{k:l-1}^{1/p}}{\widetilde{x}_{k:n-1}^{1/p}}, \qquad p_2 = \frac{\widetilde{x}_{l:n-1}^{1/p}}{\widetilde{x}_{k:n-1}^{1/p}},$$

with $p_1 + p_2 = 1$. The result to be proved becomes

$$\left[p_1 \left(\frac{\alpha_k^c - \alpha_l^c}{\widetilde{x}_{k:l-1}^{1/p}} \right)^{1-p} + p_2 \left(\frac{\alpha_l^c - \alpha_n^c}{\widetilde{x}_{l:n-1}^{1/p}} \right)^{1-p} \right]^{1/(1-p)} < \left(\frac{\alpha_k^c - \alpha_n^c}{\widetilde{x}_{k:n-1}^{1/p}} \right).$$

The power means inequality (see [36] (p. 123)) states that, for $-\infty < s < t < \infty$ and $\sum_{i=1}^n p_i = 1$,

$$\left(\sum_{i=1}^n p_i x_i^s \right)^{1/s} \leq \left(\sum_{i=1}^n p_i x_i^t \right)^{1/t}.$$

Setting $s = 1 - p$ and $t = 1$ gives, after substituting for p_1 and p_2,

$$p_1 \left(\frac{\alpha_k^c - \alpha_l^c}{\widetilde{x}_{k:l-1}^{1/p}} \right) + p_2 \left(\frac{\alpha_l^c - \alpha_n^c}{\widetilde{x}_{l:n-1}^{1/p}} \right) = \left(\frac{\alpha_k^c - \alpha_n^c}{\widetilde{x}_{k:n-1}^{1/p}} \right),$$

which proves the result. □

D.3 Total flow time—t_0-constraint applied

This section proves that $S_0^2 - S_0^1 < 0$ for cases (C_2) and (B_2).

Lemma D.4 *(i) (Case (C_2)) If $T_0^1 = T_0^2 = t_0$ and $T_1(\mu_0^2) < T_0^2 = t_0$, then $S_0^2 \leq S_0^1$.*
(ii) (Case (B_2)) If $T^1(0) < t_0$ and $T_0^2 = t_0$, then $S_0^2 \leq S_0^1$.

Proof (i) The proposition assumptions imply $0 \leq \mu_0^1 < \mu_0^2$. To show $S_0^2 < S_0^1$, write $S_0^2 - S_0^1$ as

$$S_0^2 - S_0^1 = (S_0^2 - S^1(\mu_0^2)) + (S^1(\mu_0^2) - S_0^1). \tag{D24}$$

(See Fig. 5.) The first term on the right-hand side can be expressed as (using (58))

$$S_0^2 - S^1(\mu_0^2) = (\widehat{S}^2(\mu_0^2) - \widehat{S}^1(\mu_0^2)) - \mu_0^2(T_0^2 - T^1(\mu_0^2)). \tag{D25}$$

Consider now the second term on the right-hand side of (D24). Define $t_1 = T^1(\mu_0^2)$, so that $t_1 < t_0$. (See Fig. 4F.)

The sensitivity results for Lagrange multipliers give

$$S^1(\mu_0^2) - S_0^1 = -\int_{t_0}^{t_1} \mu_1(t)\mathrm{d}t = \int_{t_1}^{t_0} \mu_1(t)\mathrm{d}t.$$

Over the range of integration $\mu_1(t) \leq \mu_0^2$. Hence,

$$S^1(\mu_0^2) - S_0^1 < \mu_0^2(t_0 - t_1). \tag{D26}$$

Thus, from (D24), (D25), and (D26),

$$\begin{aligned} S^2(\mu_2) - S^1(\mu_2) &< \widehat{S}^2(\mu_0^2) - \widehat{S}^1(\mu_0^2) - \mu_0^2(t_0 - t_1) + \mu_0^2(t_0 - t_1) \\ &= \widehat{S}^2(\mu_0^2) - \widehat{S}^1(\mu_0^2). \end{aligned}$$

From Lemma D.3, $\widehat{S}^2(\mu_0^2) - \widehat{S}^1(\mu_0^2) < 0$, proving the result.

(ii) The proposition assumptions imply $\mu_0^1 = 0$, $0 < \mu_0^2$ and $T^1(0) < T^2(0) = t_0$. (ii) is similar to (i) except that

$$S^1(\mu_0^2) - S_0^1 = -\int_{T^1(0)}^{t_1} \mu_1(t)\mathrm{d}t = \int_{t_1}^{T^1(0)} \mu_1(t)\mathrm{d}t < \mu_0^2(T^1(0) - t_1).$$

This gives

$$\begin{aligned} S_0^2 - S^1(\mu_0^2) &< \widehat{S}^2(\mu_0^2) - \widehat{S}^1(\mu_0^2) - \mu_0^2(t_0 - t_1) + \mu_0^2(T^1(0) - t_1) \\ &= \widehat{S}^2(\mu_0^2) - \widehat{S}^1(\mu_0^2) - \mu_0^2(t_0 - T^1(0)) < 0. \end{aligned}$$

The remainder of the proof is the same as in (i). □

D.4 Equal job sizes

We assume in this section a given value of μ. For equal job sizes, $\widetilde{x}_{k:l-1} = (l-k)^p x_1$, $\widetilde{x}_{l:n-1} = (n-l)^p x_1$ and $\widetilde{x}_{k:n-1} = (n-l)^p x_1$. This gives

$$\frac{\widetilde{x}_{k:l-1}}{\widetilde{x}_{k:n-1}} = \left(\frac{l-k}{n-k}\right)^p, \qquad \frac{\widetilde{x}_{l:n-1}}{\widetilde{x}_{n:n-1}} = \left(\frac{n-l}{n-k}\right)^p,$$

so that (55) becomes

$$T^2 - T^1 = (l-k)^p \frac{\alpha_k^{cp} - \alpha_l^{cp}}{(\alpha_k^c - \alpha_l^c)^p} + (n-l)^p \frac{\alpha_l^{cp} - \alpha_n^{cp}}{(\alpha_l^c - \alpha_n^c)^p} - (n-k)^p \frac{\alpha_k^{cp} - \alpha_n^{cp}}{(\alpha_k^c - \alpha_n^c)^p}. \tag{D27}$$

We prove the following.

Proposition D.1 *For an internal-split and for equal job sizes:*

(i) If $p \le 0.5$ then $T^2 - T^1 > 0$.
(ii) If $l - k \ge n - l$ then $T^2 - T^1 > 0$.

Example $p = 0.5$

To gain some insight into what happens with equal job sizes, consider the simple example of $p = 0.5$ with $a = b = 1$ and $x_j = 1$, for which $\widetilde{x}_{k:l-n} = \sqrt{2}$. In this case,

$c = 2$ and $cp = 1$, allowing the terms in (D27) to be easily factored and simplified. This gives

$$\begin{aligned} T^2 - T^1 &= \frac{1}{\sqrt{2\alpha_n + 3}} + \frac{1}{\sqrt{2\alpha_n + 1}} - \frac{2}{\sqrt{2\alpha_n + 2}} \\ &= \left(\frac{-1}{\sqrt{2\alpha_n + 2}} - \frac{-1}{\sqrt{2\alpha_n + 3}}\right) - \left(\frac{-1}{\sqrt{2\alpha_n + 1}} - \frac{-1}{\sqrt{2\alpha_n + 2}}\right). \end{aligned}$$

This is positive since $-1/\sqrt{x} - (-1/\sqrt{x+1})$ is an increasing function.

On the other hand, for the same parameter values,

$$\begin{aligned} \widehat{S}^2 - \widehat{S}^1 &= \sqrt{2\alpha_n + 3} + \sqrt{2\alpha_n + 1} - 2\sqrt{2\alpha_n + 2} \\ &= \left(\sqrt{2\alpha_n + 3} - \sqrt{2\alpha_n + 2}\right) - \left(\sqrt{2\alpha_n + 2} - 2\sqrt{2\alpha_n + 1}\right). \end{aligned}$$

This is negative since $\sqrt{x+1} - \sqrt{x}$ is a decreasing function.

Combining the above makes $S^2 - S^1 = (\widehat{S}^2 - \widehat{S}^1) - \mu(T^2 - T^1)$ negative. A major complication that occurs when p is not equal to 0.5 is that the terms cannot be easily factored in the general case. Another complication occurs when the split is unequal, that is, when $a \neq b$, and the resulting expressions become more complex.

D.4.1 $T^2 - T^1$ for equal job sizes

In the general case, for convenience, define $a = l - k$, $b = n - l$, and $A = a + b$. Thus,

$$\alpha_k = u + b + a, \qquad \alpha_l = u + b, \qquad \alpha_n = u.$$

Define also

$$z = \frac{\alpha_l}{\alpha_k} = \frac{u+b}{u+b+a}, \qquad y = \frac{\alpha_n}{\alpha_l} = \frac{u}{u+b}, \qquad x = \frac{\alpha_n}{\alpha_k} = \frac{u}{u+b+a},$$

and the functions

$$P(u) = \frac{1-u^p}{(1-u)^p}, \qquad Q(u) = P(u^c) = \frac{1-u^{cp}}{(1-u^c)^p}.$$

Lemma D.5 *The condition for $T^2 - T^1 > 0$ is equivalent to*

$$a^p Q(z) + (A-a)^p Q(y) > A^p Q(x). \tag{D28}$$

Proof Using this notation, the requirement for $T^2 - T^1 > 0$ in (D27) can be expressed as

$$a^p \frac{(u+b+a)^{cp} - (u+b)^{cp}}{((u+b+a)^c - (u+b)^c)^p} + (A-a)^p \frac{(u+b)^{cp} - u^{cp}}{((u+b)^c - u^{cp})^p}$$

$$> A^p \frac{(u+b+a)^{cp} - u^{cp}}{((u+b+a)^c - u^{cp})^p}. \tag{D29}$$

This can be expressed as (D28). □

The condition in Lemma D.5 will be shown to be true for the two cases: $p \leq 0.5$ and $a \geq b$.

D.4.2 Equal job sizes: $p \leq 0.5$

This section proves Proposition D.1(i): that $T^2 - T^1 > 0$ if $p \leq 0.5$.

Proof Rearranging (D29) gives

$$\left(\frac{a}{A}\right)^p \frac{(u+b+a)^{cp} - (u+b)^{cp}}{(u+b+a)^{cp} - u^{cp}} \left(\frac{(u+b+a)^c - u^c}{(u+b+a)^c - (u+b)^c}\right)^p + \left(\frac{A-a}{A}\right)^p \frac{(u+b)^{cp} - u^{cp}}{(u+b+a)^{cp} - u^{cp}} \left(\frac{(u+b+a)^{cp} - u^{cp}}{(u+b)^{cp} - u^{cp}}\right)^p > 1.$$

Further rearrangement gives

$$\frac{(u+b+a)^{cp} - (u+b)^{cp}}{(u+b+a)^{cp} - u^{cp}} \left(\frac{[(u+b+a)^c - u^c]/A}{[(u+b+a)^c - (u+b)^c]/a}\right)^p + \frac{(u+b)^{cp} - u^{cp}}{(u+b+a)^{cp} - u^{cp}} \left(\frac{[(u+b+a)^c - u^c]/A}{[(u+b)^c - u^c]/(A-a)}\right)^p > 1. \tag{D30}$$

Define now

$$q_z = \frac{(u+b+a)^{cp} - (u+b)^{cp}}{(u+b+a)^{cp} - u^{cp}}, \qquad q_y = \frac{(u+b)^{cp} - u^{cp}}{(u+b+a)^{cp} - u^{cp}},$$
$$r_z = \frac{a}{A} \frac{(u+b+a)^c - u^c}{(u+b+a)^c - (u+b)^c}, \qquad r_y = \frac{A-a}{A} \frac{(u+b+a)^c - u^c}{(u+b)^c - u^c}.$$

allowing (D30) to be expressed as

$$q_z r_z^p + q_y r_y^p > 1. \tag{D31}$$

The power means inequality (e.g., see [36] (p. 123)) gives, for $\sum_k p_k = 1$, $p_k, y_k \geq 0$ and $s < t$,

$$M_s = \left(\sum_k p_k y_k^s\right)^{1/s} < M_t = \left(\sum_k p_k y_k^t\right)^{1/t}.$$

For the case $t = 0$, M_0 has the value $M_0 = \prod_k x_k^{p_k}$ (obtained by taking the limit $t \to 0$). Since $q_z + q_y = 1$, application of the power means inequality gives $M_0 \leq M_p$

and so

$$r_z^{q_z} r_y^{q_y} < q_z r_z^p + q_y r_y^p.$$

It will be shown that $1 < r_z^{q_z} r_y^{q_y}$, for $p \leq 0.5$, and thus (D31) will be satisfied, thus proving the proposition.

To prove $1 < r_z^{q_z} r_y^{q_y}$, observe that for $p \leq 0.5$, $1 < c \leq 2$, and so u^c is a convex function and u^{c-1} is a concave function.

Define the following increments:

$$f = (u+b+a)^{cp} - (u+b)^{cp}, \qquad F = (u+b+a)^{cp} - u^{cp},$$
$$e = (u+b+a)^{c} - (u+b)^{c}, \qquad E = (u+b+a)^{c} - u^{c}.$$

Using these definitions,

$$r_z = \frac{a}{A}\frac{E}{e}, \qquad r_y = \frac{A-a}{A}\frac{E}{E-e}, \qquad q_z = \frac{f}{F}, \qquad q_y = \frac{F-f}{F}.$$

Therefore,

$$r_z^{q_z} r_y^{q_y} = \left(\frac{a/A}{e/E}\right)^{f/F} \left(\frac{(A-a)/A}{(E-e)/E}\right)^{1-f/F} = \frac{w^\beta (1-w)^{1-\beta}}{x^\beta (1-x)^{1-\beta}}, \tag{D32}$$

where $w = a/A$, $v = e/E$ and $\beta = f/F$. We have $f/F < a/A < e/E$, so that $\beta < w < v$. This follows since the slope of the secant of u^c between $u+b$ and $u+b+a$ is greater than the slope of the secant of u^c between u and $u+b+a$. Thus, $e/a > E/A$, and therefore $e/E > a/A$. Similarly, the slope of the secant of u^{c-1} between $u+b$ and $u+b+a$ is less than the slope of the slope of the secant of u^{c-1} between u and $u+b+a$. Thus, $f/a < F/A$, and hence, $f/F < a/A$.

Demonstrating $1 < r_z^{q_z} r_y^{q_y}$ requires that (D32) is greater than 1. To show this, observe that although β depends on v, it has the same value in the numerator and the denominator of (D32). Thus, we consider the function $u^\beta (1-u)^{1-\beta}$ as a function of u and vary u from w to v. Taking the derivative of this function gives

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}u} u^\beta (1-u)^{1-\beta} &= \beta u^{\beta-1}(1-u)^{1-\beta} - (1-\beta)u^\beta (1-u)^{-\beta} \\ &= u^{\beta-1}(1-\beta)^{-\beta}(\beta(1-u) - (1-\beta)u) \\ &= u^{\beta-1}(1-\beta)^{-\beta}(\beta - u).\end{aligned}$$

Since $\beta < w$, then, for $u \in [w, v]$, the derivative is less than zero. Hence, since $w < v$,

$$w^\beta (1-w)^{1-\beta} > x^\beta (1-v)^{1-\beta},$$

completing the proof. □

Equal job sizes: $l - k \geq n - l$

This section proves Proposition D.1(ii): that $T^2 - T^1 > 0$ if $l - k \geq n - l$.

Proof The proof of the proposition is obtained by applying Lemma D.6 followed by Lemma D.8. □

Lemma D.6 uses Jensen's inequality to obtain the conditions for (D28) to be satisfied.

Lemma D.6 *A sufficient condition for inequality* (D28) *to be satisfied is*

$$\frac{Q(y)}{b^{1-p}} > \frac{Q(z)}{a^{1-p}}. \tag{D33}$$

Proof The inequality (D29) can be expressed as

$$\begin{aligned} & a\frac{(1/z)^{cp} - 1}{a^{1-p}((1/z)^c - 1)^p} + b\frac{1 - y^{cp}}{b^{1-p}(1 - y^c)^p} \\ & > (a+b)\frac{(1/z)^{cp} - y^{cp}}{(a+b)^{1-p}((1/z)^c - y^c)^p} \\ & = (a+b)\frac{((1/z)^{cp} - 1) + (1 - y^{cp})}{(a+b)^{1-p}(((1/z)^c - 1) + (1 - y^c))^p}. \end{aligned} \tag{D34}$$

Define $p_a = a/(a+b)$, $p_b = b/(a+b)$,

$$\begin{aligned} A &= (1/z)^{cp} - 1, \qquad B = 1 - y^{cp} \\ C &= (1/z)^c - 1, \qquad D = 1 - y^c, \end{aligned}$$

to enable (D34) to be written as

$$p_a\frac{A}{a^{1-p}C^p} + p_b\frac{B}{b^{1-p}D^p} > \frac{A+B}{(a+b)^{1-p}(C+D)^p}. \tag{D35}$$

A lower bound is now obtained for $\mathscr{Q} \equiv (a+b)^{1-p}(C+D)^p$. First, rearrange $\mathscr{Q}$ as follows

$$\mathscr{Q} = \left(C(a+b)^{(1-p)/p} + D(a+b)^{(1-p)/p}\right)^p = \left(p_a C' + p_b D'\right)^p,$$

where

$$C' = C\frac{(a+b)^{1/p}}{a}, \qquad D' = D\frac{(a+b)^{1/p}}{b}.$$

Jensen's inequality gives, since the function $f(x) = x^p$ is concave,

$$\mathscr{Q} > p_a C'^p + p_b D'^p = \frac{a}{a+b}C^p\left(\frac{a+b}{a^p}\right) + \frac{b}{a+b}D^p\left(\frac{a+b}{a^p}\right)$$

$$= a^{1-p}C^p + b^{1-p}D^p.$$

Hence,

$$\frac{A+B}{a^{1-p}C^p + b^{1-p}D^p} > \frac{A+B}{(a+b)^{1-p}(C+D)^p}. \tag{D36}$$

(D35) will be proven if it can be shown that

$$p_a \frac{A}{a^{1-p}C^p} + p_b \frac{B}{b^{1-p}D^p} > \frac{A+B}{a^{1-p}C^p + b^{1-p}D^p}. \tag{D37}$$

Define $X \equiv a^{1-p}C^p$ and $Y \equiv b^{1-p}D^p$ so that (D37) becomes

$$p_a \frac{A}{X} + p_b \frac{B}{Y} > \frac{A+B}{X+Y}.$$

This can be expressed as

$$p_a + p_b \frac{U}{V} > \frac{1+U}{1+V}, \tag{D38}$$

where $U \equiv B/A$ and $V \equiv Y/X$. Expanding and simplifying (D38) gives

$$p_b \frac{U}{V} + p_a V > p_a U + p_b.$$

This can be rearranged to give $U(p_b - p_a V) > V(p_b - p_a V)$. Lemma D.7 shows that $p_b - p_a V > 0$, and therefore the condition $U = B/A > V = Y/X$ must be satisfied. This is equivalent to $B/Y > A/X$. Substituting in the definitions of the variables gives

$$\frac{1-y^{cp}}{b^{1-p}(1-y^c)^p} > \frac{1-z^{cp}}{a^{1-p}(1-z^c)^p}.$$

This is equivalent to (D33). □

Lemma D.7 $p_b - p_a V > 0$.

Proof $p_b - p_a V > 0$ is equivalent to

$$\frac{b}{a+b} - \frac{a}{a+b}\frac{b^{1-p}(1-y^c)^p}{a^{1-p}((1/z)^c - 1)^p} > 0.$$

This simplifies to

$$\frac{(1-y^c)^p}{((1/z)^c - 1)^p} < \frac{b}{a},$$

which is equivalent to

$$\frac{\sigma^{cp}_{l:n-1}}{\sigma^{cp}_{k:l-1}} < \left(\frac{n-1}{l-k}\right)^p.$$

This is satisfied by Lemma H.9(i)(a). □

Lemma D.8 shows that if $a \geq b$ and $0 < p < 1$, then the conditions for Lemma D.6 are satisfied, which completes the proof of the proposition.]

Lemma D.8 *For $a \geq b$ and $0 < p < 1$,*

$$\frac{Q(y)}{b^{1-p}} > \frac{Q(z)}{a^{1-p}}. \tag{D39}$$

Proof For the case $a = b$, proving (D39) requires proving that $Q(y) > Q(z)$. If $a = b$, then

$$z = \frac{u+b}{u+2b} > y = \frac{u}{u+b},$$

and the result follows since, by Lemma H.3(ii), Q is a decreasing function.

Now assume $a > b$. The approach will be to fix the values of b and u and increase a. In this case, the left-hand side of (D39) is constant since y only depends on u and b. It will be shown that $Q(z)/a^{1-p}$ decreases with a, and therefore the result will follow for $a > b$ since it is true $a = b$.

The derivative of $Q(z)/a^{1-p}$ is

$$\frac{\mathrm{d}}{\mathrm{d}a}\left(\frac{Q(z)}{a^{1-p}}\right) = \frac{\mathrm{d}}{\mathrm{d}a}\left(Q(z)a^{p-1}\right) = a^{p-1}\frac{\mathrm{d}}{\mathrm{d}z}Q(z)\frac{\mathrm{d}z}{\mathrm{d}a} + (p-1)a^{p-2}Q(z).$$

This will be negative if

$$a\frac{\mathrm{d}}{\mathrm{d}z}Q(z)\frac{\mathrm{d}z}{\mathrm{d}a} < (1-p)Q(z).$$

First, we have

$$\frac{\mathrm{d}z}{\mathrm{d}a} = -\frac{u+b}{(u+b+a)^2} = -\frac{1}{u+b}\left(\frac{u+b}{u+b+a}\right)^2 = -\frac{z^2}{u+b}.$$

Next, solving for a in $z = (u+b)/(u+b+a)$, gives $a = (u+b)(1-z)/z$, to give

$$a\frac{\mathrm{d}z}{\mathrm{d}a} = -\frac{z^2}{u+b}(u+b)\frac{1-z}{z} = -z(1-z).$$

Thus, we require

$$-z(1-z)\frac{\mathrm{d}}{\mathrm{d}z}Q(z) < (1-p)Q(z).$$

From Lemma H.3(i),

$$\frac{\mathrm{d}}{\mathrm{d}z}Q(z) = -(c-1)z^{-1}\left[\frac{1}{1-z^{c-1}} - \frac{1}{1-z^c}\right]Q(z).$$

Hence, what is required to be demonstrated is that

$$z(1-z)(c-1)z^{-1}\left[\frac{1}{1-z^{c-1}} - \frac{1}{1-z^c}\right]Q(z) < (1-p)Q(z),$$

which reduces to demonstrating

$$(c-1)\left[\frac{1}{1-z^{c-1}} - \frac{1}{1-z^c}\right] < \frac{(1-p)}{1-z}.$$

This can be rearranged as

$$(c-1)\frac{z^{c-1}-z^c}{(1-z^{c-1})(1-z^c)} = (c-1)\frac{z^{c-1}(1-z)}{(1-z^{c-1})(1-z^c)} < \frac{(1-p)}{1-z}.$$

Since $c = 1/(1-p)$ this becomes

$$c(c-1)z^{c-1} < \frac{(1-z^{c-1})(1-z^c)}{(1-z)^2}.$$

The left-hand side can be split to give

$$cz^{c/2}(c-1)z^{(c-1)/2} < \left(\frac{1-z^c}{1-z}\right)\left(\frac{1-z^{c-1}}{1-z}\right).$$

Lemma D.9 shows that

$$cz^{c/2} \le \frac{1-z^c}{1-z}.$$

Applying this for c, and then with c replaced by $c-1$, completes the proof of the proposition. □

Lemma D.9 *For* $0 < c$ *and* $0 < z < 1$,

$$cz^{c/2} \le \frac{1-z^c}{1-z}.$$

Proof Use $(\mathrm{d}/\mathrm{d}u)\, z^u = (\log z) z^u$ to give

$$\int_0^c (-\log z) z^u \mathrm{d}u = 1 - z^c.$$

The integral on the left-hand side can be expressed as

$$\int_0^c (-\log z) z^u \mathrm{d}u = \int_0^{c/2} (-\log z)(z^u + z^{c-u}) \mathrm{d}u,$$

to give

$$\frac{1}{1-z} \int_0^c (-\log z) z^u \mathrm{d}u = \frac{2(-\log z)}{1-z} \int_0^{c/2} \frac{z^u + z^{c-u}}{2} \mathrm{d}u.$$

By the arithmetic mean-geometric mean inequality

$$\sqrt{z^u z^{c-u}} = \sqrt{z^c} = z^{c/2} \leq \frac{z^u + z^{c-u}}{2}.$$

Hence,

$$\frac{1}{1-z} \int_0^c (-\log z) z^u \mathrm{d}u \geq \frac{-2 \log z}{1-z} \int_0^{c/2} z^{c/2} \mathrm{d}u = \frac{c(-\log z)}{1-z} z^{c/2}.$$

Since the graph of $-\log z$ is tangential to $1 - z$ at $z = 1$ and above $1 - z$ elsewhere $(z > 0)$ then $(-\log z)/(1-z) \geq 1$. Hence,

$$\frac{1 - z^c}{1-z} = \frac{1}{1-z} \int_0^c (-\log z) z^u \mathrm{d}u \geq c z^{c/2}.$$

□

D.4.3 Optimality of equal jobs sizes

This section proves that the maximum increase in the total flow time $S^2 - S^1$ (which may be negative), for a fixed value of $\widetilde{x}_{k:n-1}$, is obtained when jobs have equal sizes.

Proof (Proposition 9.4) From (57),

$$\begin{aligned} \frac{S^2 - S^1}{\widetilde{x}_{k:n-1}} &= \frac{\widetilde{x}_{k:l-1}}{\widetilde{x}_{k:n-1}} \frac{(J-k+1)\alpha_k^{cp} - (J-l+1)\alpha_l^{cp}}{\sigma_{k:l-1}^{cp}} \\ &+ \frac{\widetilde{x}_{l:n-1}}{\widetilde{x}_{k:n-1}} \frac{(J-l+1)\alpha_l^{cp} - (J-n+1)\alpha_n^{cp}}{\sigma_{l:n-1}^{cp}} \\ &- \frac{(J-k+1)\alpha_k^{cp} - (J-n+1)\alpha_n^{cp}}{\sigma_{k:n-1}^{cp}}. \end{aligned} \tag{D40}$$

Since $\widetilde{x}_{k:n-1}$ is fixed, the left-hand side of (D40) is proportional to $S^2 - S^1$. On the right-hand side of (D40), the final term is independent of the job sizes for a given value of μ. The proof is obtained by showing that the maximum value of the sum of the first two terms of the right-hand side of (D40) occurs when jobs have equal sizes. The maximum value is obtained by formulating an optimization problem. Define

$$X = \left(\frac{\widetilde{x}_{k:l-1}}{\widetilde{x}_{k:n-1}}\right), \qquad Y = \left(\frac{\widetilde{x}_{l:n-1}}{\widetilde{x}_{k:n-1}}\right),$$

and

$$A = \frac{(J-k+1)\alpha_k^{cp} - (J-l+1)\alpha_l^{cp}}{\sigma_{k:l-1}^{cp}}, \tag{D41}$$

$$B = \frac{(J-l+1)\alpha_l^{cp} - (J-n+1)\alpha_n^{cp}}{\sigma_{l:n-1}^{cp}}, \tag{D42}$$

so that the sum of the first two terms of the right-hand side of (D40) equals $AX + BY$. The goal of the optimization problem is to maximize this quantity. There are two constraints: The first constraint is derived directly from the identity for $\widetilde{x}$ given in Lemma H.7(i):

$$\left(\frac{\widetilde{x}_{k:l-1}}{\widetilde{x}_{k:n-1}}\right)^{1/p} + \left(\frac{\widetilde{x}_{l:n-1}}{\widetilde{x}_{k:n-1}}\right)^{1/p} = 1.$$

The second constraint is obtained from the upper bound on Y/X given in Lemma H.8(i):

$$\frac{Y}{X} = \frac{\left(\frac{\widetilde{x}_{l:n-1}}{\widetilde{x}_{k:n-1}}\right)}{\left(\frac{\widetilde{x}_{k:l-1}}{\widetilde{x}_{k:n-1}}\right)} = \left(\frac{\widetilde{x}_{l:n-1}}{\widetilde{x}_{k:l-1}}\right) \geq \left(\frac{n-l}{l-k}\right)^p \equiv \eta.$$

The optimization problem is

$$\begin{aligned} &\max\ AX + BY \\ &\text{s.t.} \qquad X^{1/p} + Y^{1/p} = 1 \\ &\qquad\quad \eta X - Y \leq 0. \end{aligned}$$

From Lemma D.10, the maximum value for $AX + BY$ occurs when all jobs have the same size. □

Lemma D.10 *The maximum value of $AX + BY$ occurs when*

$$X = \left(\frac{l-k}{n-k}\right)^p, \qquad Y = \left(\frac{n-l}{n-k}\right)^p,$$

which occurs when all jobs are of equal size.

Proof Lagrange multipliers are used to solve this problem. Define the Lagrangian

$$L(X, Y, \lambda, \omega) = -AX - BY - \xi(X^{1/p} + Y^{1/p} - 1) + \omega(\eta X - Y),$$

where the Lagrange multipliers are ξ and ω. Taking partial derivatives and equating to zero give

$$\frac{\partial L}{\partial X} = -A + \frac{1}{p}\xi X^{1/p-1} + \omega\eta = 0$$
$$\frac{\partial L}{\partial Y} = -B + \frac{1}{p}\xi Y^{1/p-1} - \omega = 0.$$

This has the solution

$$X = \left(\frac{(A-\omega\eta)^c}{(A-\omega\eta)^c + (B+\omega)^c}\right)^p$$
$$Y = \left(\frac{(B+\omega)^c}{(A-\omega\eta)^c + (B+\omega)^c}\right)^p.$$

If the η constraint is inactive, then $\omega = 0$ and the optimal solution is

$$X = \left(\frac{A^c}{A^c + B^c}\right)^p, \qquad Y = \left(\frac{B^c}{A^c + B^c}\right)^p.$$

However, this solution leads to a constraint violation. In particular, the solution for X and Y in this case gives

$$\frac{Y}{X} = \frac{B^{cp}}{A^{cp}}.$$

However, an inactive η constraint implies that

$$\frac{Y}{X} > \left(\frac{n-l}{l-k}\right)^p,$$

which leads to the requirement that

$$\frac{B}{A} = \left(\frac{Y}{X}\right)^{1/cp} = \eta^{1/cp} > \left(\frac{n-l}{l-k}\right)^{1/c}.$$

However, Lemma D.11 shows that $B/A < \eta^{1/cp}$, which is inconsistent with this requirement.

Hence, the η constraint must be active, and so $Y/X = \eta$. Since $X^{1/p} + (\eta X)^{1/p} = 1$, solving for X and Y gives

$$X = \left(\frac{1}{1+\eta^{1/p}}\right)^p, \qquad Y = \left(\frac{1}{1+(1/\eta)^{1/p}}\right)^p.$$

Substituting the value of η gives

$$X = \left(\frac{1}{1+\frac{n-l}{l-k}}\right)^p = \left(\frac{l-k}{n-k}\right)^p$$
$$Y = \left(\frac{1}{1+\frac{l-k}{n-l}}\right)^p = \left(\frac{n-l}{n-k}\right)^p,$$

which are the values obtained using equal job sizes. □

Lemma D.11 *For A and B defined by (D41) and (D42), respectively,*

$$\frac{B}{A} < \left(\frac{n-l}{l-k}\right)^{1-p}.$$

Proof Consider the ratio $R = \frac{B/A}{(n-l)^{1-p}/(l-k)^{1-p}} = \frac{B/(n-l)^{1-p}}{A/(l-k)^{1-p}}$. Substituting for A and B gives

$$\begin{aligned}
R &= \frac{(J-l+1)\alpha_l^{cp} - (J-n+1)\alpha_n^{cp}}{(n-l)^{1-p}(\alpha_l^c - \alpha_n^c)^p} \Bigg/ \frac{(J-k+1)\alpha_k^{cp} - (J-l+1)\alpha_l^{cp}}{(l-k)^{1-p}(\alpha_k^c - \alpha_l^c)^p} \\
&= \frac{(\alpha_l^c - \alpha_n^p) - \mu(\alpha_l^{cp} - \alpha_n^{cp})}{(n-l)^{1-p}(\alpha_l^c - \alpha_n^c)^p} \Bigg/ \frac{(\alpha_k^c - \alpha_l^c) - \mu(\alpha_k^{cp} - \alpha_l^{cp}}{(l-k)^{1-p}(\alpha_k^c - \alpha_l^c)^p} \\
&= \frac{(n-l)^p}{(\alpha_l^c - \alpha_n^c)^p}\left[\frac{\alpha_l^c - \alpha_n^c}{n-l} - \mu\frac{\alpha_l^{cp} - \alpha_n^{cp}}{n-l}\right] \Bigg/ \frac{(l-k)^p}{(\alpha_k^c - \alpha_l^c)^p}\left[\frac{\alpha_k^c - \alpha_l^c}{k-l} - \mu\frac{\alpha_k^{cp} - \alpha_l^{cp}}{l-k}\right].
\end{aligned}$$

From Lemma D.12,

$$\frac{(\alpha_l^c - \alpha_n^c) - \mu(\alpha_l^{cp} - \alpha_n^{cp})}{(\alpha_k^c - \alpha_l^c) - \mu(\alpha_k^{cp} - \alpha_l^{cp})} < \frac{\alpha_l^c - \alpha_n^c}{\alpha_k^c - \alpha_l^c}.$$

Dividing the numerators on both sides of this by $n-l$, and dividing the denominators on both sides of this by $l-k$ give

$$\begin{aligned}
&\left[\frac{\alpha_l^c - \alpha_n^c}{n-l} - \mu\frac{\alpha_l^{cp} - \alpha_n^{cp}}{n-l}\right] \Bigg/ \left[\frac{\alpha_k^c - \alpha_l^c}{l-k} - \mu\frac{\alpha_k^{cp} - \alpha_l^{cp}}{l-k}\right] \\
&< \left[\frac{\alpha_l^c - \alpha_n^c}{n-l}\right] \Bigg/ \left[\frac{\alpha_k^c - \alpha_l^c}{l-k}\right].
\end{aligned}$$

Hence, it is required to prove that, noting $n-l = \alpha_l - \alpha_n$ and $l-k = \alpha_k - \alpha_l$,

$$R < \frac{(n-l)^p}{(\alpha_l^c - \alpha_n^c)^p}\left[\frac{\alpha_l^c - \alpha_n^c}{n-l}\right] \Bigg/ \frac{(l-k)^p}{(\alpha_k^c - \alpha_l^c)^p}\left[\frac{\alpha_k^c - \alpha_l^c}{l-k}\right]$$

$$= \left(\frac{\alpha_l^c - \alpha_n^c}{\alpha_l - \alpha_n}\right)^{1-p} \Bigg/ \left(\frac{\alpha_k^c - \alpha_l^c}{\alpha_k - \alpha_l}\right)^{1-p}. \tag{D43}$$

Applying the mean value theorem to the function $f(u) = u^c$ gives

$$\frac{\alpha_l^c - \alpha_n^c}{\alpha_l - \alpha_n} = c\theta^{c-1} \quad \text{and} \quad \frac{\alpha_k^c - \alpha_l^c}{\alpha_k - \alpha_l} = c\phi^{c-1},$$

where $\alpha_n < \theta < \alpha_l$ and $\alpha_l < \phi < \alpha_k$. Since $\theta < \phi$, the right-hand side of (D43) equals

$$\left(\frac{c\theta^{c-1}}{c\phi^{c-1}}\right)^{1-p} = \left(\left(\frac{\theta}{\phi}\right)^{c-1}\right)^{1-p},$$

which is less than 1, and so, $R < 1$. □

Lemma D.12

$$\frac{(\alpha_l^c - \alpha_n^c) - \mu(\alpha_l^{cp} - \alpha_n^{cp})}{(\alpha_k^c - \alpha_l^c) - \mu(\alpha_k^{cp} - \alpha_l^{cp})} \leq \frac{\alpha_l^c - \alpha_n^c}{\alpha_k^c - \alpha_l^c}. \tag{D44}$$

Proof For $\mu = 0$, the result is trivial. For $0 < \mu$, (D44) is equivalent to

$$1 - \mu\frac{\alpha_l^{cp} - \alpha_n^{cp}}{\alpha_l^c - \alpha_n^c} < 1 - \mu\frac{\alpha_k^{cp} - \alpha_l^{cp}}{\alpha_k^c - \alpha_l^c},$$

which simplifies to

$$\frac{\alpha_k^{cp} - \alpha_l^{cp}}{\alpha_k^c - \alpha_l^c} < \frac{\alpha_l^{cp} - \alpha_n^{cp}}{\alpha_l^c - \alpha_n^c}.$$

Applying the mean value theorem to the function $f(u) = u^p$, gives the required inequality as

$$p\phi^{p-1} < p\theta^{p-1}, \tag{D45}$$

where $\alpha_n^c < \theta < \alpha_l^c$ and $\alpha_l^c < \phi < \alpha_k^c$. Rearranging (26) gives $1 < (\phi/\theta)^{1-p}$. Since $\theta < \phi$, the result follows. For $\mu > 0$, the inequality in (D44) is strict. □

Appendix E Constraint Qualification

It is well known that a minimum point need not satisfy the KKT conditions: this can occur if the dimension of the set of linearized feasible directions is greater than the dimension of the tangent cone at a point (e.g., see [35]). However, if a suitable

constraint qualification is satisfied, then a minimum point must satisfy the KKT conditions. The most common constraint qualification is the linear independence constraint qualification (LICQ), which is satisfied at a point if the set of equality constraint gradients and active inequality gradients are linearly independent. The paper shows that LICQ will be satisfied, except for some exceptional cases described in the following.

For the optimization problem in (13), the equality constraints are

$$h_n(\boldsymbol{\tau}, \boldsymbol{\theta}) = \sum_{j=n}^{J} \theta_{j,n} - \psi_n, \qquad n = 1, \cdots, J,$$
$$H_j(\boldsymbol{\tau}, \boldsymbol{\theta}) = \sum_{n=1}^{j} s_{j,n}\tau_n - x_j, \qquad j = 1, \cdots, J,$$

and the inequality constraints are

$$g_0(\boldsymbol{\tau}, \boldsymbol{\theta}) = \sum_{n=1}^{J} \tau_n - t_0,$$
$$g_n(\boldsymbol{\tau}, \boldsymbol{\theta}) = -\tau_n, \qquad n = 1, \cdots, J.$$

The gradients of these constraint functions are, respectively, as follows. (In the vector notation below, the variables corresponding to nonzero values are positioned above the gradient vector elements according to the partial differentiation order: $(\tau_1, \cdots, \tau_J, \theta_{1,1}, \cdots, \theta_{J,1}, \theta_{21}, \cdots, \theta_{J,2}, \cdots, \theta_{J,J})$.)

$$\nabla h_n(\boldsymbol{\tau}, \boldsymbol{\theta}) = \left(\frac{\cdots\ \theta_{n,n}\ \cdots\ \theta_{J,n}\ \cdots}{1\ \cdots\ 1} \right)^T,$$
$$\nabla H_j(\boldsymbol{\tau}, \boldsymbol{\theta}) = \left(\frac{\tau_1\ \cdots\ \tau_j\ \cdots\ \theta_{j,1}\ \cdots\ \theta_{j,j}\ \cdots}{s_{j,1}\quad s_{j,j}\quad \tau_1 s'_{j,1}\ \cdots\ \tau_j s'_{j,j}\ \cdots} \right)^T,$$
$$\nabla g_0(\boldsymbol{\tau}, \boldsymbol{\theta}) = \left(\frac{\tau_1\ \cdots\ \tau_J\ \cdots}{1\quad 1} \right)^T,$$
$$\nabla g_n(\boldsymbol{\tau}, \boldsymbol{\theta}) = \left(\frac{\cdots\ \tau_n\ \cdots}{-1} \right)^T.$$

Define $\mathscr{A} \subseteq \{0, 2, \cdots, J\}$ as the set of active inequality constraints (the index 1 is not included since it is assumed that $\tau_1 > 0$). LICQ requires that the vectors

$$\nabla H_1, \cdots, H_J, \nabla h_1, \cdots, \nabla h_J, \{\nabla g_i, i \in \mathscr{A}\} \tag{E46}$$

are linearly independent; that is, the only solution to

$$u_1 \nabla H_1 + \cdots u_J \nabla H_J + v_1 \nabla h_1 + \cdots + v_J \nabla h_J + \sum_{i \in \mathscr{A}} a_i \nabla g_i = \mathbf{0}^T \tag{E47}$$

is for all coefficients $\{u_i\}, \{v_i\}, \{a_i\}$ to be equal to 0. Define

$$A = (\nabla H_1, \cdots, \nabla H_J, \nabla h_1, \cdots, \nabla h_J), \qquad B = (\nabla g_i)_{i \in \mathscr{A}},$$

whereby (E47) can be expressed as[11]

$$Ay + Bz = \mathbf{0}^T, \tag{E48}$$

where $y = (u_1, \cdots, u_J, v_1, \cdots, v_J)^T$ and $z = (a_i, i \in \mathscr{A})^T$. Now, decompose A as

$$A = \begin{pmatrix} A_1 \\ A_0 \end{pmatrix},$$

where A_1 comprises the first J rows of A (corresponding to $\boldsymbol{\tau}$) and A_0 comprises the remaining rows. Since only the first J rows of B are nonzero it can be written as

$$B = \begin{pmatrix} B_1 \\ \mathbf{0} \end{pmatrix}.$$

with A_1 and B_1 having the same number of rows. With this decomposition, Eq. (E48) becomes

$$\begin{pmatrix} A_1 \\ A_0 \end{pmatrix} y = - \begin{pmatrix} B_1 \\ \mathbf{0} \end{pmatrix} z.$$

LICQ is satisfied if the only solution to this equation is $y = \mathbf{0}^T$ and $z = \mathbf{0}^T$. Solution of this equation requires that y belong to the null space of A_0, where $y = y_0$ is in the null space of A_0 if

$$A_0 y_0 = \mathbf{0}. \tag{E49}$$

Thus, the condition for LICQ is that if y_0 is in the null space of A and

$$A_1 y_0 = B_1 z \tag{E50}$$

[11] To reduce notational complexity, the dimensions of the null vectors and the vectors of all ones in the following equations are not explicitly stated, but can be easily inferred from the equation context.

then y_0 must equal $\mathbf{0}^T$. The null space of A_0 is obtained by converting A_0 to reduced row echelon form. The following considers the case for $J = 3$.

$$A = \begin{pmatrix} s_{11} & s_{21} & s_{31} & 0\,0\,0 \\ 0 & s_{22} & s_{32} & 0\,0\,0 \\ 0 & 0 & s_{33} & 0\,0\,0 \\ \tau_1 s'_{11} & 0 & 0 & 1\,0\,0 \\ 0 & \tau_1 s'_{21} & 0 & 1\,0\,0 \\ 0 & 0 & \tau_1 s'_{31} & 1\,0\,0 \\ 0 & \tau_2 s'_{22} & 0 & 0\,1\,0 \\ 0 & 0 & \tau_2 s'_{32} & 0\,1\,0 \\ 0 & 0 & \tau_3 s'_{33} & 0\,0\,1 \end{pmatrix},$$

to give

$$A_1 = \begin{pmatrix} s_{11} & s_{21} & s_{31} & 0\,0\,0 \\ 0 & s_{22} & s_{32} & 0\,0\,0 \\ 0 & 0 & s_{33} & 0\,0\,0 \end{pmatrix}$$

and

$$A_0 = \begin{pmatrix} \tau_1 s'_{11} & 0 & 0 & 1\,0\,0 \\ 0 & \tau_1 s'_{21} & 0 & 1\,0\,0 \\ 0 & 0 & \tau_1 s'_{31} & 1\,0\,0 \\ 0 & \tau_2 s'_{22} & 0 & 0\,1\,0 \\ 0 & 0 & \tau_2 s'_{32} & 0\,1\,0 \\ 0 & 0 & \tau_3 s'_{33} & 0\,0\,1 \end{pmatrix}. \tag{E51}$$

The reduced row echelon form for A_0 can be computed to be:

$$\begin{pmatrix} 1\,0\,0 & \frac{1}{\tau_1 s'_{11}} & 0\,0 \\ 0\,1\,0 & \frac{1}{\tau_1 s'_{21}} & 0\,0 \\ 0\,0\,1 & \frac{1}{\tau_1 s'_{31}} & 0\,0 \\ 0\,0\,0 & -\frac{\tau_2 s'_{22}}{\tau_1 s'_{21}} + \frac{\tau_2 s'_{32}}{\tau_1 s'_{31}} & 0\,0 \\ 0\,0\,0 & -\frac{\tau_2 s'_{32}}{\tau_1 s'_{31}} & 1\,0 \\ 0\,0\,0 & -\frac{\tau_3 s'_{33}}{\tau_1 s'_{31}} & 0\,1 \end{pmatrix}. \tag{E52}$$

If $-\frac{\tau_2 s'_{22}}{\tau_1 s'_{21}} + \frac{\tau_2 s'_{32}}{\tau_1 s'_{31}} \neq 0$, then this matrix can be further reduced to become

$$\begin{pmatrix} 1&0&0&0&0&0\\ 0&1&0&0&0&0\\ 0&0&1&0&0&0\\ 0&0&0&1&0&0\\ 0&0&0&0&1&0\\ 0&0&0&0&0&1 \end{pmatrix}.$$

The dimension of the null space is 0 and the only solution of (E49) is $y_0 = \mathbf{0}$. If $-\frac{\tau_2 s'_{22}}{\tau_1 s'_{21}} + \frac{\tau_2 s'_{32}}{\tau_1 s'_{31}} = 0$ the null space has dimension 1 and consists of solutions of the form

$$v_1 \begin{pmatrix} -\frac{1}{\tau_1 s'_{11}}\\ -\frac{1}{\tau_1 s'_{21}}\\ -\frac{1}{\tau_1 s'_{31}}\\ 1\\ \frac{\tau_2 s'_{32}}{\tau_1 s'_{31}}\\ \frac{\tau_3 s'_{33}}{\tau_1 s'_{31}} \end{pmatrix}. \tag{E53}$$

The two solutions can be combined so that the first case is obtained by only allowing $v_1 = 0$. Observe that it is the second possibility that occurs at the optimal allocation points since, for the $J = 3$ case,

$$\begin{aligned} -\frac{\tau_2 s'_{22}}{\tau_1 s'_{21}} + \frac{\tau_2 s'_{32}}{\tau_1 s'_{31}} &= -\frac{\tau_2}{\tau_1}\left(\left(\frac{\theta_{22}}{\theta_{21}}\right)^{p-1} - \left(\frac{\theta_{32}}{\theta_{31}}\right)^{p-1}\right)\\ &= -\frac{\tau_2}{\tau_1}\left(\left(\frac{\sigma_2^c/\alpha_2^c}{\sigma_2^c/\alpha_1^c}\right)^{p-1} - \left(\frac{\sigma_3^c/\alpha_2^c}{\sigma_3^c/\alpha_1^c}\right)^{p-1}\right) = 0. \end{aligned}$$

This gives

$$A_1 y_0 = v_1 \begin{pmatrix} s_{11}&s_{21}&s_{31}&0&0&0\\ 0&s_{22}&s_{32}&0&0&0\\ 0&0&s_{33}&0&0&0 \end{pmatrix} \begin{pmatrix} -\frac{1}{\tau_1 s'_{11}}\\ -\frac{1}{\tau_1 s'_{21}}\\ -\frac{1}{\tau_1 s'_{31}}\\ 1\\ \frac{\tau_2 s'_{32}}{\tau_1 s'_{31}}\\ \frac{\tau_3 s'_{33}}{\tau_1 s'_{31}} \end{pmatrix}$$

$$= -\frac{v_1}{\tau_1}\begin{pmatrix} \frac{s_{11}}{s'_{11}} + \frac{s_{21}}{s'_{21}} + \frac{s_{31}}{s'_{31}} \\ \frac{s_{22}}{s_{21}}\frac{s_{21}}{s'_{21}} + \frac{s_{32}}{s_{31}}\frac{s_{31}}{s'_{31}} \\ \frac{s_{33}}{s_{31}}\frac{s_{31}}{s'_{31}} \end{pmatrix} = -\frac{v_1}{p\tau_1}\begin{pmatrix} \theta_{11}+\theta_{21}+\theta_{31} \\ \frac{s_{22}}{s_{21}}\theta_{21} + \frac{s_{32}}{s_{31}}\theta_{31} \\ \frac{s_{33}}{s_{31}}\theta_{31} \end{pmatrix}.$$

Different cases for this equation are now considered for different inequality constraint activations.

E.1 All inequality constraints inactive

If all inequality constraints are inactive then (E46) reduces to

$$u_1\nabla H_1 + \cdots u_J\nabla H_J + v_1\nabla h_1 + \cdots + v_J\nabla h_J = \mathbf{0}^T.$$

The LICQ condition, (E50), then becomes

$$A_1 y_0 = -\frac{v_1}{p\tau_1}\begin{pmatrix} \theta_{11}+\theta_{21}+\theta_{31} \\ \frac{s_{22}}{s_{21}}\theta_{21} + \frac{s_{32}}{s_{31}}\theta_{31} \\ \frac{s_{33}}{s_{31}}\theta_{31} \end{pmatrix} = \mathbf{0}^T.$$

For v_1 nonzero, $A_1 y_0$ is also nonzero, and this equation will not hold. Hence, LICQ is satisfied.

E.2 t_0-constraint active, no τ_n-constraints active

Suppose now that the t_0-constraint is active and that no τ_n-constraints are active. In this case,

$$B_1 z = a_0\begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix},$$

so that the condition for LICQ, (E50), becomes

$$A_1 y_0 = -\frac{v_1}{p\tau_1}\begin{pmatrix} \theta_{11}+\theta_{21}+\theta_{31} \\ \frac{s_{22}}{s_{21}}\theta_{21} + \frac{s_{32}}{s_{31}}\theta_{31} \\ \frac{s_{33}}{s_{31}}\theta_{31} \end{pmatrix} = \begin{pmatrix} a_0 \\ a_0 \\ a_0 \end{pmatrix}.$$

The elements $\theta_{11}+\theta_{21}+\theta_{31} = 1$ and $(\frac{s_{33}}{s_{31}}\theta_{31} = \theta_{31}^{p-1})$ are not equal, except for $\theta_{31} = 1$ or $p = 1$. The case $\theta_{31} = 1$ cannot occur since this means job 3 will be completed first, which violates the completion order assumptions. For $p = 1$, the elements of $A_1 y_0$ are equal and therefore LICQ is not satisfied.

E.3 t_0-constraint and one or more τ_n-constraints active

Suppose now that the t_0-constraint is active and that one or more τ_n-constraints are active. In this case,

$$B_1 z = a_0 \begin{pmatrix} 1 \\ 1 \\ 1 \\ \cdots \\ 1 \end{pmatrix} + a_2 \begin{pmatrix} 0 \\ -1 \\ 0 \\ \cdots \\ 0 \end{pmatrix} + \cdots + a_J \begin{pmatrix} 0 \\ 0 \\ 0 \\ \cdots \\ -1 \end{pmatrix} = \begin{pmatrix} a_0 \\ a_0 - a_2 \\ a_0 - a_3 \\ \cdots \\ a_0 - a_J \end{pmatrix}.$$

where $a_j = 0$ for inactive constraints. The LICQ condition (E50) becomes

$$A_1 y_0 = -\frac{v_1}{p\tau_1} \begin{pmatrix} \theta_{11} + \theta_{21} + \theta_{31} \\ \frac{s_{22}}{s_{21}}\theta_{21} + \frac{s_{32}}{s_{31}}\theta_{31} \\ \frac{s_{33}}{s_{31}}\theta_{31} \end{pmatrix} = \begin{pmatrix} a_0 \\ a_0 - a_2 \\ a_0 - a_3 \end{pmatrix} = w.$$

If 2 or more of the τ_n-constraints are inactive, then at least two elements of w will be equal. However, for $p < 1$, $A_1 y_0$ cannot be equal to such a vector, and so LICQ is satisfied. If $p = 1$, then the elements of $A_1 y_0$ are identical, but $A_1 y_0$ cannot be equal to w since at least two τ_n-constraints are active, and the elements of w are not all the same. Therefore, LICQ is satisfied in this case.

However, if only one of the τ_n-constraints is inactive, then LICQ cannot be satisfied since given any nonzero y_0, there exists $a_0, a_1, \cdots, a_J$, not all zero, such that $w = A_1 y_0$, because the vectors

$$\begin{pmatrix} 1 \\ 1 \\ 1 \\ \cdots \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ -1 \\ 0 \\ \cdots \\ 0 \end{pmatrix}, \cdots, \begin{pmatrix} 0 \\ 0 \\ 0 \\ \cdots \\ -1 \end{pmatrix},$$

constitute a basis for $\mathbb{R}^J$. In this case, however, all jobs are completed simultaneously: The feasible set consists of a single point, and the value of the objective function is equal to $\widetilde{x}_{1:J}$.

E.4 t_0-constraint inactive and one or more τ_n-constraints active

Suppose now that the t_0-constraint is inactive and that one or more τ_n-constraints are active. In this case,

$$B_1 z = a_2 \begin{pmatrix} 0 \\ -1 \\ 0 \\ \cdots \\ 0 \end{pmatrix} + \cdots + a_J \begin{pmatrix} 0 \\ 0 \\ 0 \\ \cdots \\ -1 \end{pmatrix} = \begin{pmatrix} 0 \\ -a_2 \\ -a_3 \\ \cdots \\ -a_J \end{pmatrix},$$

where $a_j = 0$ for inactive constraints. The LICQ condition, (E50), becomes

$$A_1 y_0 = -\frac{v_1}{p\tau_1} \begin{pmatrix} \theta_{11} + \theta_{21} + \theta_{31} \\ \frac{s_{22}}{s_{21}}\theta_{21} + \frac{s_{32}}{s_{31}}\theta_{31} \\ \frac{s_{33}}{s_{31}}\theta_{31} \end{pmatrix} = v_1 \begin{pmatrix} 0 \\ -a_2 \\ -a_3 \end{pmatrix}.$$

For $v_1 \neq 0$, all elements of the left-hand side ($A_1 y_0$) are nonzero. Hence, this equation cannot be satisfied, and so LICQ is satisfied.

The general case is similar; however, due to space considerations, it is not shown.

Appendix F Solution for $p = 1$

Proof (Proposition 6.7)

If $p = 1$, then $s_{j,n} = \theta_{j,n}$ and Eq. (15) becomes

$$x_j = \sum_{n=1}^{j} \theta_{j,n}\tau_n, \qquad j = 1, \cdots, J. \tag{F54}$$

(i) Summing (F54) gives

$$\sum_{j=1}^{J} x_j = \sum_{j=1}^{J}\sum_{n=1}^{j} \theta_{j,n}\tau_n = \sum_{n=1}^{J} \tau_n \sum_{j=n}^{J} \theta_{j,n} = \sum_{n=1}^{J} \tau_n = T.$$

This applies regardless of the completion pattern. (Alternatively, since $\sum_{j=1}^{n} s_{j,n} = 1$, the system is work-conserving and the result follows using, for example, Eq. (1) of [15]).

(ii) To obtain the total flow time, first express (F54) as

$$x_j = \sum_{n=1}^{j-1} \theta_{j,n}\tau_n + \theta_{j,j}\tau_j.$$

Multiplying x_j by $(J - j + 1)$ and summing gives

$$\begin{aligned} &\sum_{j=1}^{J} (J - j + 1)x_j \\ &= \sum_{j=1}^{J} (J - j + 1) \sum_{n=1}^{j-1} \theta_{j,n}\tau_n + \sum_{j=1}^{J} (J - j + 1)\theta_{j,j}\tau_j \\ &= \sum_{n=1}^{J} \tau_n \sum_{j=n+1}^{J} (J - j + 1)\theta_{j,n} + \sum_{n=1}^{J} (J - n + 1)\tau_n \left(1 - \sum_{j=n+1}^{J} \theta_{j,n}\right) \end{aligned}$$

$$= \sum_{n=1}^{J}(J-n+1)\tau_n - \sum_{n=1}^{J}\tau_n \sum_{j=n+1}^{J}\theta_{j,n}(j-n)$$
$$= \sum_{n=1}^{J}(J-n+1)\tau_n - \sum_{j=1}^{J}\sum_{n=1}^{j}\tau_n\theta_{j,n}(j-n).$$

Thus,

$$S = \sum_{n=1}^{J}(J-n+1)\tau_n = \sum_{j=1}^{J}(J-j+1)x_j + \sum_{j=1}^{J}\sum_{n=1}^{j}\tau_n\theta_{j,n}(j-n).$$

Since $j-n \geq 0$, $\tau_n > 0$ and $\theta_{j,n} \geq 0$, S has the minimum value of

$$\sum_{j=1}^{J}(J-j+1)x_j,$$

which is achieved when $\theta_{j,n}(j-n) = 0$. This implies that $\theta_{j,n} = 0$ $(j > n)$ and $\theta_{j,j} = 1$.

(iii) Suppose there is a single simultaneous completion epoch at the end of stage m, and k is the number of jobs that complete there. To compute the total flow time, proceeding as in the case of individual completions gives,

$$S = \sum_{n=1}^{J}(J-n+1)\tau_n = \sum_{j=1}^{J}(J-j+1)x_j + A,$$

where $A = \sum_{j=1}^{J}\sum_{n=1}^{j}(j-n)\tau_n\theta_{j,n}$. Decompose A as follows:

$$A = \sum_{j=1}^{m-1}\sum_{n=1}^{j}(j-n)\tau_n\theta_{j,n} + \sum_{j=m}^{m+k-1}\sum_{n=1}^{j}(j-n)\tau_n\theta_{j,n} + \sum_{j=m+k}^{J}\sum_{n=1}^{j}(j-n)\tau_n\theta_{j,n}$$
$$= A_{1:m-1} + A_{m:m+k-1} + A_{m+k:J}.$$

The term $A_{m:m+k-1}$ can be further decomposed as

$$A_{m:m+k-1} = \sum_{j=m}^{m+k-1}\sum_{n=1}^{j}(j-n)\tau_n\theta_{j,n}$$
$$= \sum_{j=m}^{m+k-1}\sum_{n=1}^{j}(m-n)\tau_n\theta_{j,n} + \sum_{j=m}^{m+k-1}\sum_{n=1}^{j}(j-m)\tau_n\theta_{j,n}$$

$$= \sum_{j=m}^{m+k-1} \sum_{n=1}^{m} (m-n)\tau_n \theta_{j,n} + \sum_{j=m}^{m+k-1} (j-m) \sum_{n=1}^{m} \tau_n \theta_{j,n}$$
$$= B_{m:m+k-1} + C_{m:m+k-1}.$$

The second last line was obtained by noting that since k jobs complete simultaneously at index m, $\tau_n = 0$ for $m+1 \le n \le m+k-1$.

We now show $C_{m:m+k-1} = \sum_{j=m}^{m+k-1} (j-m)x_j$. We have, for $m \le j \le m+k-1$, that (using $\tau_{m+1} = \tau_{m+k-1} = 0$ again)

$$x_j = \sum_{n=1}^{m} \theta_{j,n} \tau_n.$$

Multiplying by $m-j$ and summing from m to $m+k-1$ gives

$$\sum_{j=m}^{m+k-1} (j-m)x_j = \sum_{j=m}^{m+k-1} \sum_{n=1}^{m} (j-m)\tau_n \theta_{j,n} = C_{m:m+k-1}.$$

Hence,

$$S = \sum_{j=1}^{J} (J-j+1)x_j + A_{1:m-1} + B_{m:m+k-1} + A_{m+k:J} + \sum_{j=m}^{m+k-1} (j-m)x_j$$
$$= \sum_{j=1}^{m-1} (J-j+1)x_j + \sum_{j=m}^{m+k-1} (J-m+1)x_j + \sum_{j=m+k}^{J} (J-j+1)x_j$$
$$+ A_{1:m-1} + B_{m:m+k-1} + A_{m+k:J}.$$

The only terms containing $\theta_{j,n}$ are the final three terms. The smallest total flow time occurs when each of these three terms equals 0. This gives the desired result (51). The final three terms are 0 when:

- $A_{1:m-1} = 0$ if $(j-n)\theta_{j,n} = 0$ $(j = 1, \cdots, m-1, n = 1, \cdots, j)$. This implies $\theta_{j,n} = 0$ $(n < j)$. $\theta_{j,j}$ can nonzero.
- $A_{m+k:J} = 0$ if $(j-n)\theta_{j,n} = 0$ $(j = m+k, \cdots, J, n = 1, \cdots, j)$. This implies $\theta_{j,n} = 0$ $(n < j)$. $\theta_{j,j}$ can nonzero.
- $B_{m+k:m+k-1} = 0$, if $(m-n)\theta_{j,n} = 0$ $(j = m, \cdots, m+k-1, n = 1, \cdots, m)$. This implies $\theta_{j,n} = 0$ $(n < m)$. For $m+1 \le n \le m-k+1$, the allocations can be ignored since $\tau_n = 0$. For $n = m$, $\theta_{j,m}$ $(j = m, \cdots, m+k-1)$ can be nonzero.

Combining the above, the cases where $\theta_{j,n}$ equals 0 are (a) $n < m, n < j \le J$, (b) $n = m, m+k \le j$ and (c) $m+k \le n, n < j \le J$. The cases where $\theta_{j,n}$ are nonzero are the following.

(a) For the single completion epochs ($n < m$ and $m+k \le n$), since $\sum_{j=n}^{J} \theta_{j,n} = 1$ and $\theta_{j,n} = 0$ $(n < j)$, $\theta_{n,n} = 1$.

(b) For the multiple completion epoch ($n = m$), jobs m to $m + k - 1$ receive all the cores in that stage: $\sum_{j=m}^{m+k=1} \theta_{j,m} = 1$. The cores are divided between these jobs so that jobs $m \le j \le m + k - 1$ complete simultaneously. The following shows that the cores are allocated in proportion to the job size. For these jobs

$$x_j = \sum_{n=1}^{m} \theta_{j,n}\tau_n = \sum_{n=1}^{m-1} \theta_{j,n}\tau_n + \theta_{j,m}\tau_m = \theta_{j,m}\tau_m.$$

Hence,

$$\sum_{j=m}^{m+k-1} x_j = \sum_{j=m}^{m+k-1} \theta_{j,m}\tau_m = \tau_m.$$

This gives

$$\theta_{j,m} = \frac{x_j}{\tau_m} = \frac{x_j}{x_m + \cdots + x_{m+k-1}}.$$

(iv) The single simultaneous completion epoch analysis of (iii) extends naturally to the analysis of multiple simultaneous completion epochs. For each such epoch, the simultaneously completing jobs can be treated as a single job of size equal to the sum of the sizes of the jobs completing, and cores are allocated in proportion to the job size. This gives the following, with the notation extending naturally from the proof of (iii).

$$\sum_{n=1}^{J}(J - n + 1)\tau_n = \sum_{j=1}^{J}(J - j + 1)x_j + \sum_i \sum_{j=m_i}^{m_i+k_i-1}(j - m_i)x_j + \sum_i A_{m_{i-1}:m_i-1} + \sum_i A_{m_i:m_i+k_i-1} + A_{m_i+k_i:J}.$$

The derivation now follows the single simultaneous completion case. Details are omitted. □

Appendix G Solution for $p = 0$

If $p = 0$, then $s_{j,n} = 1$. In this case, the equality constraint (15) becomes $\sum_{n=1}^{j} \tau_n = x_j$, to give $T = \sum_{n=1}^{J} \tau_n = x_J$. Hence, $S = \sum_{n=1}(J - n)\tau_j = \sum_{j=1}^{J}\sum_{n=1}^{j} \tau_n = \sum_{j=1}^{J} x_j$. Thus, all allocations θ generate the same makespan and total flow time.

Appendix H Lemmas

This section presents supporting technical lemmas that provide properties of the functions used and bounds for expressions involving $\widetilde{x}$ and α_n. The insights these lemmas provide is that many of the results depend on the growth rates of different sequences, in particular, for the sequences defined by $(J+n-1)$, (α_n), (σ_n), (x_n), the powers of the terms in the sequences and the difference sequences generated by them.

H.1 Lemmas involving properties of functions

Lemma H.1 *For $0 < p < 1$, let g be the function defined by*

$$g(z) = \frac{1 - az^p}{(1-z)^p}.$$

(i) The derivative of $g(z)$ is

$$g'(z) = -\frac{p(a - z^{1-p})}{(1-z)^{p+1}z^{1-p}}.$$

(ii) $g(z)$ is a decreasing function of z for $z < a^c$, and $g(z)$ is an increasing function of z for $z > a^c$.

Proof (i) We have

$$g(z) = \frac{1}{(1-z)^p} - \frac{a}{(1/z-1)^p} = (1-z)^{-p} - a((1/z)-1)^{-p}.$$

Hence,

$$\begin{aligned}\frac{\mathrm{d}g}{\mathrm{d}z} &= p(1-z)^{-p-1} - a(-p)((1/z)-1)^{-p-1}(-1/z^2) \\ &= \frac{p}{(1-z)^{1+p}} - \frac{ap}{(1/z-1)^{1+p}z^2} \\ &= -\left(\frac{ap}{(1-z)^{1+p}z^{1-p}} - \frac{p}{(1-z)^{1+p}}\right).\end{aligned}$$

(ii) The result follows immediately from (i) and $c = 1/(1-p)$.

□

Lemma H.2 *The derivative of $P(z) = \frac{1-z^p}{(1-z)^p}$ is*

$$\frac{d}{dz}P(z) = -\frac{p(1-z^{1-p})}{(1-z)^{1+p}z^{1-p}} = -\frac{p(1-z^{p-1})}{(1-z)^{1+p}}.$$

(ii) For $0 < p < 1$, (a) $P(z)$ is decreasing; (b) $\lim_{z\to 1} P(z) = 0$; (c) $0 < P(z) < 1$.

Proof (i) Follows from Lemma H.1(i) by setting $a = 1$.
(ii) (a) This follows from Lemma H.1(ii). (b) Using l'Hôpital's rule gives

$$\lim_{z\to 1} P(z) = \lim_{z\to 1} \frac{-pz^{p-1}}{p(1-z)^{p-1}} = -\lim_{z\to 1} p(1-z)^{1-p} = 0.$$

(c) This follows from (a), $g(0) = 1$ and $P(z)$ decreasing.

□

Lemma H.3 *(i) The derivative of* $Q(z) = \frac{1-z^{cp}}{(1-z^c)^p}$ *is*

$$\frac{d}{dz}Q(z) = \frac{cpz^{c-1}(1-z^{-1})}{(1-z^c)^{1+p}} = -(c-1)z^{-1}\left[\frac{1}{1-z^{c-1}} - \frac{1}{1-z^c}\right]Q(z).$$

(ii) $Q(z)$ *is a decreasing function for* $0 < z < 1$.

Proof (i)

$$\frac{\mathrm{d}}{\mathrm{d}z}Q(z) = P'(z^c)cz^{c-1} = \frac{cpz^{c-1}(1-z^{c(p-1)})}{(1-z^c)^{1+p}} = \frac{cpz^{c-1}(1-z^{-1})}{(1-z^c)^{1+p}}.$$

Rearranging this gives

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}z}Q(z) &= \frac{cpz^{c-1}(1-z^{-1})}{(1-z^c)(1-z^{cp})}\frac{1-z^{cp})}{(1-z^c)^p} = -\frac{(c-1)z^{c-2}(1-z)}{(1-z^c)(1-z^{c-1})}Q(z)\\ &= -(c-1)z^{-1}\left[\frac{1}{1-z^{c-1}} - \frac{1}{1-z^c}\right]Q(z).\end{aligned}$$

(ii) This follows immediately from (i).

□

H.2 Lemmas involving series coefficients

Lemma H.4 *Suppose* $\mu \geq 0$ *and*

$$f(\mu) = \frac{\alpha_n^{cp} - \alpha_{n+k}^{cp}}{(\alpha_n^c - \alpha_{n+k}^c)^p}.$$

Then (a) $f(\mu)$ *is a decreasing function of* μ*; (b)* $1 > f(\mu) > 0$*; (c)* $\lim_{\mu\to\infty} f(\mu) = 0$.

Proof $f(\mu)$ can be written as $P(z)$ where $z = (\alpha_{n+k}/\alpha_n)^c$. Since $\alpha_{n+k} = \alpha_n - k$, $z < 1$ for $\mu \geq 0$. Taking the derivative of $f(\mu)$ gives

$$\frac{\mathrm{d}f(\mu)}{\mathrm{d}\mu} = \frac{\mathrm{d}P(z)}{\mathrm{d}z}\frac{\mathrm{d}z}{\mathrm{d}\mu}.$$

Applying Lemma H.2 gives $\mathrm{d}P/\mathrm{d}z \le 0$ for $z < 1$. Next, since $z = ((\alpha_n - k)/\alpha_n)^c = (1 - k/\alpha_n)^c$ is an increasing function of α_n, and α_n is an increasing function of μ, $\mathrm{d}z/\mathrm{d}\mu > 0$. Therefore, $f(\mu)$ is a decreasing function of μ ($\mu \ge 0$). (b) and (c) follow from Lemma H.2(ii). □

Lemma H.5 *Suppose $\mu \ge 0$ and*

$$h(\mu) = \frac{(J-n+1)\alpha_n^{cp} - (J-n+1-k)\alpha_{n+k}^{cp}}{(\alpha_n^c - \alpha_{n+k}^c)^p}.$$

Then (a) $h(\mu)$ is an increasing function of μ; (b) $h(\mu) \to \infty$ ($\mu \to \infty$).

Proof (a) $h(\mu)$ can be written as

$$g(z) = \frac{1 - az^p}{(1-z)^p}, \tag{H55}$$

where $z = (\alpha_{n+k}/\alpha_n)^c$ and $a = (J-n+1-k)/(J-n+1)$. Taking the derivative of $h(\mu)$ gives

$$\frac{\mathrm{d}h(\mu)}{\mathrm{d}\mu} = \frac{\mathrm{d}g(z)}{\mathrm{d}z}\frac{\mathrm{d}z}{\mathrm{d}\mu}.$$

Applying Lemma H.1 gives $\mathrm{d}g/\mathrm{d}z \ge 0$ for $z \ge a^c = ((J-n+1-k)/(J-n+1))^c$. Since $(J-n+1+\mu-k)/(J-n+1+\mu)$ increases for $\mu \ge 0$ and achieves its smallest value of a^c for $\mu = 0$, this implies $\mathrm{d}g(z)/\mathrm{d}z > 0$ for $\mu \ge 0$. As in Lemma H.4, $\mathrm{d}z/\mathrm{d}\mu > 0$. Therefore, $h(\mu)$ is an increasing function of μ.

(b) As $\mu \to \infty$, then $z \to 1$. Since $a < 1$, the result follows from (H55). □

Lemma H.6 *Suppose $n > 0$, $k > 0$ and $n + k \le J$, and*

$$f(\mu) = \alpha_n^c - \alpha_{n+k}^c = (J-n+1+\mu)^c - (J-n-k+1+\mu)^c.$$

Then, $f(n)$ is an increasing function of μ, and $f(n)$ is a decreasing function of n. The latter implies $\sigma_r > \sigma_s$, for $r < s$.

Proof First, $\alpha_n = J - n + \mu$ and $\alpha_{n+k} = \alpha_n - k$. Differentiating with respect to μ gives

$$\frac{\mathrm{d}}{\mathrm{d}\mu} f(\mu) = \frac{\mathrm{d}}{\mathrm{d}\alpha_n}(\alpha_n^c - (\alpha_n - k)^c)\frac{\mathrm{d}\alpha_n}{\mathrm{d}\mu} = c\left(\alpha_n^{c-1} - (\alpha_n - k)^{c-1}\right) > 0,$$

since $c > 1$ and $\mathrm{d}\alpha_n/\mathrm{d}\mu > 0$. Differentiation with respect to n is similar, except that the derivative $\mathrm{d}\alpha_n/\mathrm{d}n$ is negative. □

H.3 Lemmas involving $\widetilde{x}_n$

Lemma H.7 *The follow identities involving $\widetilde{x}$ are used in the paper:*

$$(i) \quad \widetilde{x}_{k:l-1}^{1/p} + \widetilde{x}_{l:n-1}^{1/p} = \widetilde{x}_{k:n-1}^{1/p},$$
$$(ii) \quad \left(\frac{\widetilde{x}_{k:n-1}}{\widetilde{x}_{l:n-1}}\right)^{1/p} = \left(\frac{\widetilde{x}_{k:l-1}}{\widetilde{x}_{l:n-1}}\right)^{1/p} + 1,$$
$$(iii) \quad \left(\frac{\widetilde{x}_{k:n-1}}{\widetilde{x}_{k:l-1}}\right)^{1/p} = 1 + \left(\frac{\widetilde{x}_{l:n-1}}{\widetilde{x}_{k:l-1}}\right)^{1/p}.$$

Proof (i)

$$\widetilde{x}_{k:n-1}^{1/p} = \left(x_k^{1/p} + \cdots + x_{l-1}^{1/p}\right) + \left(x_l^{1/p} + \cdots + x_{n-1}^{1/p}\right) = \widetilde{x}_{k:l-1}^{1/p} + \widetilde{x}_{l:n-1}^{1/p}.$$

(ii)

$$\left(\frac{\widetilde{x}_{k:n-1}}{\widetilde{x}_{l:n-1}}\right)^{1/p} = \frac{\widetilde{x}_{k:l-1}^{1/p} + \widetilde{x}_{l:n-1}^{1p}}{\widetilde{x}_{l:n-1}^{1/p}} = \left(\frac{\widetilde{x}_{k:l-1}}{\widetilde{x}_{l:n-1}}\right)^{1/p} + 1.$$

(iii)

$$\left(\frac{\widetilde{x}_{k:n-1}}{\widetilde{x}_{k:l-1}}\right)^{1/p} = \frac{\widetilde{x}_{k:l-1}^{1/p} + \widetilde{x}_{l:n-1}^{1p}}{\widetilde{x}_{k:l-1}^{1/p}} = 1 + \left(\frac{\widetilde{x}_{l:n-1}}{\widetilde{x}_{k:l-1}}\right)^{1/p}.$$

□

Lemma H.8 *For $\{x_j\}$ in SJF order and $k < l < n$,*

$$(i) \quad \left(\frac{n-l}{l-k}\right)^p \le \frac{\widetilde{x}_{l:n-1}}{\widetilde{x}_{k:l-1}}, \qquad (ii) \quad \left(\frac{n-k}{l-k}\right)^p \le \frac{\widetilde{x}_{k:n-1}}{\widetilde{x}_{k:l-1}},$$
$$(iii) \quad \left(\frac{n-l}{n-k}\right)^p \le \frac{\widetilde{x}_{l:n-1}}{\widetilde{x}_{k:n-1}}.$$

Proof (i) Applying the definitions of $\widetilde{x}_{k:l-1}$ and $\widetilde{x}_{l:n-1}$ gives

$$\left(\frac{\widetilde{x}_{l:n-1}}{\widetilde{x}_{k:l-1}}\right)^{1/p} = \frac{x_l^{1/p} + \cdots + x_{n-1}^{1/p}}{x_k^{1/p} + \cdots + x_{l-1}^{1/p}} \ge \frac{x_l^{1/p} + \cdots + x_l^{1/p}}{x_{l-1}^{1/p} + \cdots + x_{l-1}^{1/p}}$$
$$= \frac{(n-l)x_l^{1/p}}{(l-k)x_{l-1}^{1/p}} \ge \frac{n-l}{l-k}.$$

(ii) From Lemma H.7(iii),

$$\left(\frac{\widetilde{x}_{k:n-1}}{\widetilde{x}_{k:l-1}}\right)^{1/p} = 1 + \left(\frac{\widetilde{x}_{l:n-1}}{\widetilde{x}_{k:l-1}}\right)^{1/p} \geq 1 + \frac{n-l}{l-k} = \frac{n-k}{l-k}.$$

where (i) has been used to obtain the final inequality.

(iii) From Lemma H.7(ii),

$$\left(\frac{\widetilde{x}_{k:n-1}}{\widetilde{x}_{l:n-1}}\right)^{1/p} = 1 + \left(\frac{\widetilde{x}_{k:l-1}}{\widetilde{x}_{l:n-1}}\right)^{1/p} \leq 1 + \frac{l-k}{n-l} = \frac{n-k}{n-l}.$$

□

Inequalities will become equalities in Lemma H.8 if jobs have equal sizes.

H.4 Lemmas involving ratios of terms containing α_n

This section presents lemmas involving ratios of terms containing α_n.

Lemma H.9 *Suppose $k < l < m < n$. (i) For $n \leq J$:*

$$(a)\ \frac{\sigma^{cp}_{l:n-1}}{\sigma^{cp}_{k:l-1}} < \left(\frac{n-l}{l-k}\right)^p, \qquad (b)\ \frac{\sigma^{cp}_{k:n-1}}{\sigma^{cp}_{k:l-1}} < \left(\frac{n-k}{l-k}\right)^p,$$
$$(c)\ \frac{\sigma^{cp}_{l:n-1}}{\sigma^{cp}_{k:n-1}} < \left(\frac{n-l}{n-k}\right)^p.$$

(ii) For $n = J+1$, $\sigma^c_{k:J} = \alpha^c_k$ and $\sigma^c_{l:J} = \alpha^c_l$,

$$(a)\ \frac{\alpha^{cp}_l}{\sigma^{cp}_{k:l-1}} < \left(\frac{\alpha_l}{l-k}\right)^p, \qquad (b)\ \frac{\alpha^{cp}_k}{\sigma^{cp}_{k:l-1}} < \left(\frac{\alpha_k}{l-k}\right)^p,$$
$$(c)\ 1 < \left(\frac{\alpha_k}{\alpha_l}\right)^p < \frac{\sigma^{cp}_{k:J}}{\sigma^{cp}_{l:J}}.$$

Proof (i)(a) Applying the definitions of $\sigma^c_{k:l-1}$ and $\sigma^c_{l:n-1}$ gives the following.

$$\frac{\sigma^c_{l:n-1}}{\sigma^c_{k:l-1}} = \frac{\alpha^c_l - \alpha^c_n}{\alpha^c_k - \alpha^c_l} = \frac{(\alpha^c_l - \alpha^c_{l+1}) + \cdots + (\alpha^c_{n-1} - \alpha^c_n)}{(\alpha^c_k - \alpha^c_{k+1}) + \cdots + (\alpha^c_{l-1} - \alpha^c_l)}.$$

Since, by Lemma H.6, $\alpha^c_m - \alpha^c_{m+1}$ is decreasing in m,

$$\frac{\sigma^c_{l:n-1}}{\sigma^c_{k:l-1}} < \frac{(\alpha^c_l - \alpha^c_{l+1}) + \cdots + (\alpha^c_l - \alpha^c_{l+1})}{(\alpha^c_{l-1} - \alpha^c_l) + \cdots + (\alpha^c_{l-1} - \alpha^c_l)} = \frac{(n-l)(\alpha^c_l - \alpha^c_{l+1})}{(l-k)(\alpha^c_{l-1} - \alpha^c_l)} < \frac{n-l}{l-k}.$$

(i)(b) Using the result of (i)(a),

$$\frac{\sigma^c_{k:n-1}}{\sigma^c_{k:l-1}} = \frac{\alpha^c_k - \alpha^c_n}{\alpha^c_k - \alpha^c_l} = \frac{(\alpha^c_k - \alpha^c_l) + (\alpha^c_l - \alpha^c_n)}{\alpha^c_k - \alpha^c_l} = 1 + \frac{\alpha^c_l - \alpha^c_n}{\alpha^c_k - \alpha^c_l}$$
$$< 1 + \frac{n-l}{l-k} = \frac{n-k}{l-k}.$$

(i)(c) Using the result of (i)(a),

$$\frac{\sigma^c_{k:n-1}}{\sigma^c_{l:n-1}} = \frac{\alpha^c_k - \alpha^c_n}{\alpha^c_l - \alpha^c_n} = \frac{(\alpha^c_k - \alpha^c_l) + (\alpha^c_l - \alpha^c_n)}{\alpha^c_l - \alpha^c_n}$$
$$= \frac{\alpha^c_k - \alpha^c_l}{\alpha^c_l - \alpha^c_n} + 1 > \frac{l-k}{n-l} + 1 = \frac{n-k}{n-l}.$$

(ii)(a) First,

$$\frac{\alpha^c_l}{\alpha^c_k - \alpha^c_l} = \frac{\alpha^c_l - (\alpha_l - \alpha_l))^c}{\alpha_l} \frac{l-k}{(\alpha_l + (l-k))^c - \alpha^c_l} \left(\frac{\alpha_l}{l-k}\right).$$

The mean value theorem applied to the function $f(x)$ states that: For given $y_1 < y_2$ then $(f(y_2) - f(y_1))/(y_2 - y_2) = f'(y')$ where $y_1 \le y' \le y_2$. Applying the mean value theorem to the function $f(y) = (y-a)^c$ gives, for a given constant $a > 0$,

$$\frac{y^c - (y-a)^c}{a} = c(y - \phi a)^{c-1},$$

for some $0 \le \phi \le 1$. Applying the mean value theorem to the function $f(y) = y^c$ gives, for a given constant $b > 0$,

$$\frac{(y+b)^c - y^c}{b} = c(y + \phi' b)^{c-1},$$

for some $0 \le \phi' \le 1$. Applying this to $y = \alpha_l$, $a = \alpha_l$ and $b = l - k$ gives, for some $0 \le \phi, \phi' \le 1$,

$$\frac{\alpha^c_l}{\alpha^c_k - \alpha^c_l} = \frac{(\alpha_l - \phi\alpha_l)^{c-1}}{(\alpha_l + \phi'(l-k))^{c-1}} \left(\frac{\alpha_l}{l-k}\right) < \frac{\alpha_l}{l-k}.$$

(b) Using the result of (ii)(a),

$$\frac{\alpha^c_k}{\alpha^c_k - \alpha^c_l} = \frac{(\alpha^c_k - \alpha^c_l) + \alpha^c_l}{\alpha^c_k - \alpha^c_l} = 1 + \frac{\alpha^c_l}{\alpha^c_k - \alpha^c_l}$$
$$< 1 + \frac{\alpha_l}{l-k} = \frac{\alpha_l + l - k}{l-k} = \frac{\alpha_k}{l-k}.$$

(c)

$$\frac{\alpha_k^c}{\alpha_l^c} = \frac{\alpha_k^{c-1}}{\alpha_l^{c-1}} \left(\frac{\alpha_k}{\alpha_l}\right) > \frac{\alpha_k}{\alpha_l} > 1.$$

□

Lemma H.10 *Suppose $k < l < n$. For $n \leq J$:*

$$(i)\ \ \frac{\widetilde{x}_{k:l-1}}{\sigma_{k:l-1}^{cp}} < \frac{\widetilde{x}_{l:n-1}}{\sigma_{l:n-1}^{cp}}, \qquad (ii)\ \ \frac{\widetilde{x}_{k:l-1}}{\sigma_{k:l-1}^{cp}} < \frac{\widetilde{x}_{k:n-1}}{\sigma_{k:n-1}^{cp}},$$
$$(iii)\ \ \frac{\widetilde{x}_{k:n-1}}{\sigma_{k:n-1}^{cp}} < \frac{\widetilde{x}_{l:n-1}}{\sigma_{l:n-1}^{cp}}.$$

Proof (i)

$$\frac{\sigma_{l:n-1}^{cp}}{\sigma_{k:l-1}^{cp}} < \left(\frac{n-l}{l-k}\right)^p \leq \frac{\widetilde{x}_{l:n-1}^{cp}}{\widetilde{x}_{k:l-1}}.$$

(ii)

$$\frac{\sigma_{k:n-1}^{cp}}{\sigma_{k:l-1}^{cp}} < \left(\frac{n-k}{l-k}\right)^p \leq \frac{\widetilde{x}_{k:n-1}^{cp}}{\widetilde{x}_{k:l-1}}.$$

(iii)

$$\frac{\sigma_{l:n-1}^{cp}}{\sigma_{k:n-1}^{cp}} \leq \left(\frac{n-l}{n-k}\right)^p \leq \frac{\widetilde{x}_{l:n-1}^{cp}}{\widetilde{x}_{k:n-1}}.$$

□

Lemma H.11 *For $k < l < n$,*

$$\frac{v_l}{u_l} = \frac{(J-l+1)\alpha_l^{cp} - (J-n+1)\alpha_n^{cp}}{\alpha_l^{cp} - \alpha_n^{cp}} \tag{H56}$$
$$< \frac{v_k}{u_k} = \frac{(J-k+1)\alpha_k^{cp} - (J-l+1)\alpha_m^{cp}}{\alpha_k^{cp} - \alpha_l^{cp}}. \tag{H57}$$

Proof The left-hand side of (H57) can be expressed as

$$\frac{(J-l+1)\alpha_l^{cp} - (J-l+1)\alpha_n^{cp}}{\alpha_l^{cp} - \alpha_n^{cp}} + \frac{(J-l+1)\alpha_n^{cp} - (J-n+1)\alpha_n^{cp}}{\alpha_l^{cp} - \alpha_n^{cp}}$$
$$= (J-l+1) + \frac{(n-l)\alpha_n^{cp}}{\alpha_l^{cp} - \alpha_n^{cp}}.$$

Along similar lines, the right-hand side of (H57) can be expressed as

$$(J-k+1)+\frac{(l-k)\alpha_l^{cp}}{\alpha_k^{cp}-\alpha_l^{cp}}.$$

Thus, the proof of the lemma involves showing that

$$(J-l+1)+\frac{(n-l)\alpha_n^{cp}}{\alpha_l^{cp}-\alpha_n^{cp}}<(J-k+1)+\frac{(l-k)\alpha_l^{cp}}{\alpha_k^{cp}-\alpha_l^{cp}}.$$

This can be transformed to give

$$\frac{n-l}{l-k}<\left(\frac{\alpha_l^{cp}-\alpha_n^{cp}}{\alpha_k^{cp}-\alpha_l^{cp}}\right)\frac{\alpha_k^{cp}}{\alpha_n^{cp}}.$$

Hence, what is required to be shown is that

$$\left(\frac{(\alpha_l^{cp}-\alpha_n^{cp})/(n-l)}{(\alpha_k^{cp}-\alpha_l^{cp})/(l-k)}\right)\frac{\alpha_k^{cp}}{\alpha_n^{cp}}>1.$$

Applying the mean value theorem as in Lemma H.9 gives

$$\left(\frac{cp(\alpha_l-\phi(n-l))^{cp-1}}{cp(\alpha_l+\phi'(l-k))^{cp-1}}\right)\frac{\alpha_k^{cp}}{\alpha_n^{cp}}>1.$$

The complication here is that $cp-1=c-2>-1$. To circumvent this problem, first, expand the previous inequality to be

$$\left(\frac{(\alpha_l-\phi(n-l))^{cp-1}}{(\alpha_l+\phi'(l-k))^{cp-1}}\right)\left(\frac{(\alpha_l+(l-k))^{cp}}{(\alpha_l-(n-l))^{cp}}\right)>1.$$

This can be rewritten as

$$\left(\frac{(\alpha_l-\phi(n-l))^{cp}}{(\alpha_l+\phi'(l-k))^{cp}}\right)\left(\frac{(\alpha_l+(l-k))^{cp}}{(\alpha_l-(n-l))^{cp}}\right)>\frac{(\alpha_l-\phi(n-l))}{(\alpha_l+\phi'(l-k))}.$$

Switching around the denominators on the left-hand side now gives

$$\left(\frac{(\alpha_l-\phi(n-l))^{cp}}{(\alpha_l-(n-l))^{cp}}\right)\left(\frac{(\alpha_l+(l-k))^{cp}}{(\alpha_l+\phi'(n-l))^{cp}}\right)>\frac{(\alpha_l-\phi(n-l))}{(\alpha_l+\phi'(l-k))}.$$

Both terms in the product on the left-hand side are greater than 1, whereas the right-hand side is less than 1. Hence, the result is proved. □

Acknowledgements The author acknowledges the thoughtful suggestions of the anonymous Reviewer and the Editor for improving the presentation of the results given in the paper.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions.

References

1. Gartner: Gartner says cloud will be the centerpiece of new digital experiences. https://www.gartner.com/en/newsroom/press-releases/2021-11-10-gartner-says-cloud-will-be-the-centerpiece-of-new-digital-experiences (2021/11/10) (2021)
2. Caznet: What is the difference between a data centre and cloud computing? https://caznet.com.au/what-is-the-difference-between-a-data-centre-and-cloud-computing/ (accessed 2021/11/22) (2021)
3. Harchol-Balter, M.: Open problems in queueing theory inspired by datacenter computing. Queue. Syst. **97**(1), 3–37 (2021)
4. Zhang, W., Fang, V., Panda, A., Shenker, S.: Kappa: A programming framework for serverless computing. In: Proceedings of the 11th ACM Symposium on Cloud Computing, pp. 328–343 (2020)
5. Hill, M.D., Marty, M.R.: Amdahl's law in the multicore era. Computer **41**(7), 33–38 (2008)
6. Berg, B., Vesilo, R., Harchol-Balter, M.: heSRPT: Optimal parallel scheduling of jobs with known sizes. arXiv preprint arXiv:1903.09346 (2019)
7. Berg, B., Vesilo, R., Harchol-Balter, M.: heSRPT: Parallel scheduling to minimize mean slowdown. Perform. Eval. **144**, 102147 (2020)
8. Brutlag, J.: Speed Matters for Google Web Search. Mountain View, CA, USA (2009)
9. Hong, C.-Y., Caesar, M., Godfrey, P.B.: Finishing flows quickly with preemptive scheduling. ACM SIGCOMM Comput. Commun. Rev. **42**(4), 127–138 (2012)
10. Chen, W., Rao, J., Zhou, X.: Preemptive, low latency datacenter scheduling via lightweight virtualization. In: 2017 {USENIX} Annual Technical Conference ({USENIX}{ATC} 17), pp. 251–263 (2017)
11. Wilson, C., Ballani, H., Karagiannis, T., Rowtron, A.: Better never than late: Meeting deadlines in datacenter networks. ACM SIGCOMM Comput. Commun. Rev. **41**(4), 50–61 (2011)
12. Hoff, T.: Latency is everywhere and it costs you sales - How to crush it. http://highscalability.com/latency-everywhere-and-it-costs-you-sales-how-crush-it (2009)
13. Pinedo, M.L.: Scheduling: Theory, Algorithms, and Systems, 6th edn. Springer, Gewerbestrasse (2022)
14. Weiss, G.: Scheduling and Control of Queueing Networks, vol. 14. Cambridge University Press, Cambridge (2021)
15. Ayesta, U.: A unifying conservation law for single-server queues. J. Appl. Probab. **44**(4), 1078–1087 (2007)
16. Aalto, S., Penttinen, A., Lassila, P., Osti, P.: On the optimal trade-off between SRPT and opportunistic scheduling. In: Proceedings of the ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems, pp. 185–196 (2011)
17. Netto, M.A., Calheiros, R.N., Rodrigues, E.R., Cunha, R.L., Buyya, R.: HPC cloud for scientific and business applications: Taxonomy, vision, and research challenges. ACM Comput. Surv. (CSUR) **51**(1), 1–29 (2018)
18. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Commun. ACM **51**(1), 107–113 (2008)
19. Jiang, W., Ravi, V.T., Agrawal, G.: A map-reduce system with an alternate api for multi-core environments. In: 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, pp. 84–93 (2010). IEEE.

20. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al: Tensorflow: A system for large-scale machine learning. In: 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16), pp. 265–283 (2016)
21. Demirović, D., Skejić, E., Šerifović–Trbalić, A.: Performance of some image processing algorithms in tensorflow. In: 2018 25th International Conference on Systems, Signals and Image Processing (IWSSIP), pp. 1–4 (2018)
22. Mo, Y.J., Kim, J., Kim, J.-K., Mohaisen, A., Lee, W.: Performance of deep learning computation with TensorFlow software library in GPU-capable multi-core computing platforms. In: 2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN), pp. 240–242 (2017)
23. Bienia, C., Kumar, S., Singh, J.P., Li, K.: The parsec benchmark suite: Characterization and architectural implications. In: Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques, pp. 72–81 (2008)
24. Amdahl, G.M.: Validity of the single processor approach to achieving large scale computing capabilities. In: Proceedings of the April 18-20, 1967, Spring Joint Computer Conference, pp. 483–485 (1967)
25. Zhan, X., Bao, Y., Bienia, C., Li, K.: Parsec3.0: A multicore benchmark suite with network stacks and SPLASH-2X. ACM SIGARCH Comput. Architect. News **44**(5), 1–16 (2017)
26. Fareghzadeh, N., Seyyedi, M.A., Mohsenzadeh, M.: Toward holistic performance management in clouds: taxonomy, challenges and opportunities. J. Supercomput. **75**(1), 272–313 (2019)
27. Wu, Z., Fu, L.: Optimizing job completion time with fairness in large-scale data centers. Fut. Gen. Comput. Syst. **114**, 563–573 (2021)
28. Jia, R., Yang, Y., Grundy, J., Keung, J., Li, H.: A deadline constrained preemptive scheduler using queuing systems for multi-tenancy clouds. In: 2019 IEEE 12th International Conference on Cloud Computing (CLOUD), pp. 63–67 (2019)
29. Berg, B., Harchol-Balter, M., Moseley, B., Wang, W., Whitehouse, J.: Optimal resource allocation for elastic and inelastic jobs. arXiv preprint arXiv:2005.09745 (2020)
30. Lin, S.-H., Paolieri, M., Chou, C.-F., Golubchik, L.: A model-based approach to streamlining distributed training for asynchronous sgd. In: 2018 IEEE 26th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), pp. 306–318 (2018)
31. Feitelson, D.G., Rudolph, L., Schwiegelshohn, U., Sevcik, K.C., Wong, P.: Theory and practice in parallel job scheduling. In: Workshop on Job Scheduling Strategies for Parallel Processing, pp. 1–34 (1997). Springer
32. Cera, M.C., Georgiou, Y., Richard, O., Maillard, N., Navaux, P.O.: Supporting malleability in parallel architectures with dynamicCPUSETs mapping and dynamic MPI. In: International Conference on Distributed Computing and Networking, pp. 242–257 (2010). Springer
33. Berg, B., Dorsman, J.-P., Harchol-Balter, M.: Towards optimality in parallel job scheduling. In: Abstracts of the 2018 ACM International Conference on Measurement and Modeling of Computer Systems, pp. 116–118 (2018)
34. Boyd, S., Vandenberghe, L.: Convex Analysis. Cambridge University Press, Cambridge (2006)
35. Nocedal, J., Wright, S.J.: Numerical Optimization, 2nd edn. Springer, Berlin (2006)
36. Steele, J.M.: The Cauchy-Schwarz Master Class: an Introduction to the Art of Mathematical Inequalities. Cambridge University Press, Cambridge (2004)