Greedy "Exploitation" Is Close to Optimal on Node-Heterogeneous Clusters*

Arnold L. Rosenberg

Colorado State University, Fort Collins, CO 80523, USA
Northeastern University, Boston, MA 02115, USA
rsnbrg@cs.umass.edu

Abstract. The Cluster-Exploitation Problem (CEP) challenges a *master* computer to schedule a "borrowed" node-heterogeneous cluster $\mathcal{C}$ of *worker* computers in a way that maximizes the amount of work that $\mathcal{C}$'s computers complete within a fixed time period. This challenge is heightened by the fact that "completing" work requires $\mathcal{C}$'s computers to return results from their work to the master. It has been known for some time that the greedy *LIFO* protocol, which orchestrates $\mathcal{C}$'s computers to finish working in the *opposite* of their starting order, *does not* solve the CEP optimally; in fact, the *FIFO* protocol, which has $\mathcal{C}$'s computers finish working in the *same* order as they start, *does* solve the CEP optimally (over sufficiently long time periods). That said, the LIFO protocol has features (aside from its intuitive appeal) that would make it attractive to implement when solving the CEP—as long as its solution to the problem was not too far from optimal. This paper shows this to be the case. Specifically:

1. The LIFO protocol provides approximately optimal solutions to the CEP, in the following sense. For every cluster $\mathcal{C}$, there is a fixed fraction $\varphi_{\mathcal{C}} > 0$ *that does not depend on how heterogeneous cluster* $\mathcal{C}$ *is* (as measured by the relative speeds of its fastest and slowest computers) such that $\mathcal{C}$ completes at least the fraction $\varphi_{\mathcal{C}}$ as much work under the LIFO protocol as under the optimal FIFO protocol.

Our analysis of the CEP uncovers an unexpected property of the LIFO protocol:

2. In common with the FIFO protocol, the LIFO protocol's work production is independent of the order in which the master supplies work to the workers—no matter what the relative speeds of the workers are.

Within the literature of *divisible load scheduling*, the CEP follows the master-worker paradigm under the "single-port with no overlap" model.

Keywords: Scheduling divisible workload; Worksharing; Heterogeneous cluster.

1 Introduction

A *master* computer C_0 has a large uniform computational workload of independent tasks. It has temporary access to a cluster[1] $\mathcal{C}$ comprising n *worker* computers,

* Research supported in part by US NSF Grant CNS-0905399.

[1] We call $\mathcal{C}$ a "cluster" for convenience: the C_i may be geographically dispersed and more diverse in power than that term usually connotes.

$C_1, \ldots, C_n$, that may differ in computing power: each C_i can execute a unit of work in ρ_i time units—and these n *computing rates* can be very different. All $n + 1$ computers intercommunicate across a single network that they access with uniform cost. (Our model is, thus, *node-heterogeneous* and *link-homogeneous*.) The elements of C_0's workload are *divisible,* in the sense that each can be subdivided at will to accommodate the differing computing rates of $\mathcal{C}$'s computers. The Cluster-Exploitation Problem (CEP, for short) is a simple scheduling problem under which C_0 has access to $\mathcal{C}$'s computers for some predetermined "lifespan" of L time units, during which:

1. For each $i \in \{1, \ldots, n\}$, in some order, C_0 sends some "personalized" number, w_i, units of work to each worker computer C_i, in a single message;
2. Each worker computer executes the work it receives and returns its results to C_0, in a single message.

The challenge is to orchestrate the preceding process so that $\mathcal{C}$'s computers collectively complete as much work as possible during the L time units—while ensuring that *at most one intercomputer message is in transit in the network at any step.* A unit of work is *completed* once C_0 has sent it to some C_i, and C_i has executed the unit and returned results to C_0. Within the literature of *divisible load scheduling*, the CEP follows the master-worker paradigm under the "single-port with no overlap" model. We call a schedule for solving the CEP a *worksharing protocol.* The significance of the CEP stems from the demonstration in [2] that the optimal work-production of a cluster $\mathcal{C}$ depends *only* on $\mathcal{C}$'s vector of *computing rates*, $\langle \rho_1, \ldots, \rho_n \rangle$, which we call $\mathcal{C}$'s *heterogeneity profile*.

What is the optimal schedule for solving the CEP on an n-computer cluster $\mathcal{C}$? We cite a hyperbolic instance of the CEP to garner some intuition. For convenience, let us index $\mathcal{C}$'s computers in nonincreasing order of speed, so that $\rho_1 \leq \cdots \leq \rho_n$. (Recall that each ρ_i is the time to complete one unit of work, so a smaller ρ-value means a faster computer.) Now (here's the hyperbole) say that $\mathcal{C}$'s computers are *very* different in speed: each C_i is 10^{10} times faster than C_{i-1}: formally, $\rho_{i+1} = 10^{10}\rho_i$. It is "intuitively obvious" that the optimal solution to this instance of the CEP is for the master C_0 to proceed as follows:

1. Saturate C_1 ($\mathcal{C}$'s fastest computer) with work that takes it L time units to complete.
2. Recursively solve the CEP for $C_2, \ldots, C_n$, for the lifespan determined by the portion of the L time units when neither C_n's work nor its results are in transit.

This "obviously optimal" *greedy* solution embodies the *LIFO* protocol (with workers served in order of speed): $\mathcal{C}$'s computers are orchestrated to finish working (and return results to C_0) in the *opposite* of the order in which they are served. The first surprise concerning the CEP appeared in [10], where it was shown that the LIFO protocol *does not* solve the CEP optimally. The second surprise was the demonstration in [2] that, over sufficiently long lifespans, the *FIFO* protocol, which has $\mathcal{C}$'s computers finish working (and return results to C_0) in the *same* order as they start, *does* solve the CEP optimally.[2] (The non-idle intervals in Fig. 1 suggest the origin of the names "LIFO," "FIFO.")

The results in [2,10] apparently lessen the importance of the LIFO protocol—but this view may be shortsighted. This paper revisits the LIFO protocol and shows it to have

[2] Simulations in [1] suggest that "sufficiently long" lifespans have quite modest lengths.

advantages that may make it an attractive protocol for the CEP—even though it is not optimal. We show that, in addition to having a simple recursive structure, which makes the protocol easy to specify, implement, and analyze:

> *The LIFO protocol is approximately as powerful as the FIFO protocol in solving the* CEP (Theorem 4).
> Specifically, for every cluster $\mathcal{C}$, there is a fixed fraction $\varphi_{\mathcal{C}} > 0$ *that does not depend on how heterogeneous cluster $\mathcal{C}$ is* (as measured by the relative speeds of its fastest and slowest computers) such that $\mathcal{C}$ completes at least the fraction $\varphi_{\mathcal{C}}$ as much work under the LIFO protocol as under the optimal FIFO protocol.

On the road to this result, we uncover a rather surprising property of the LIFO protocol.

> *The LIFO protocol's work production is independent of the order in which the master C_0 supplies work to the workers in cluster $\mathcal{C}$* (Theorem 3).
> Even in our extreme example, $\mathcal{C}$'s computers complete the same amount of work when the slowest worker (C_n) is allocated the longest time slot as when the fastest one (C_1) is. (This independence can be derived from results in [4], but the proof we present is quite elegant and may have further application.)

Related work. Employing a model that is very similar to ours, [3] derives efficient optimal or near-optimal schedules for the four variants of the CEP for clusters $\mathcal{C}$ that correspond to the four paired answers to the questions: "Do tasks produce nontrivial-size results?" "Is $\mathcal{C}$'s network pipelined?" For those variants that are **NP**-Hard, *near*-optimality is the most that one can expect to achieve efficiently—and this is what [3] achieves. More details on this and related work are available in the survey [11]. One finds in [12] a study of heterogeneity in computing that is based on the fact (from [2]) that optimal solutions to the CEP for a cluster $\mathcal{C}$ depend only on $\mathcal{C}$'s heterogeneity profile; this study explores features of $\mathcal{C}$'s profile that determine its work-completion rate and that give one cluster a higher rate than another. A variant of the CEP in which clusters are *node-homogeneous* but *link-heterogeneous* is studied in [5]; in that setting, the FIFO protocol loses its advantage over the LIFO protocol: neither protocol dominates the other. Less directly related to our study is the large body of work that studies the scheduling of "divisible workloads." While parts of sources such as [6,7,9] and their kin share our interest in the CEP, their focus on tasks that do not produce measurable output that must be returned to the master allows much simpler algorithmics; e.g., the FIFO and LIFO protocols coincide in their model.

2 Formal Details

We adapt the model of [8], which is the basis of [2,5,10,12].

The computing environment. The *master* computer C_0's workload is composed of work units that are identical in size and complexity.[3] *The tasks' (common) complexity can be an arbitrary function of their (common) size.* Our model posits that the cost of transmitting work grows *linearly* with the total amount of work performed. This allows us to *measure both time and message-length in the same units as work.*

[3] "Size" refers to specification length, "complexity" to computation time.

This linear communication model ignores *fixed* transmission costs—the end-to-end latency of a message's first packet and the per-message set-up overhead—because their impacts fade over long lifespans. Thus, we replace the *affine* communication model of [2,8] with a *linear* model. We justify this simplification via two facts that hold asymptotically, i.e., over "sufficiently long life-spans." (a) For the CEP, the linear and affine models coincide asymptotically. (b) The optimality result from [2] that motivates the current study (cited as our Theorem 1) holds only asymptotically.

For $i \in \{1, \ldots, n\}$, a *worker* computer C_i that belongs to the cluster $\mathcal{C}$ of interest can execute one unit of work in ρ_i time units; this ρ-value is C_i's *computing rate*. For convenience, we normalize the computing rates of $\mathcal{C}$'s computers, so that if $\mathcal{C}$'s *(heterogeneity) profile* is $\langle \rho_1, \ldots, \rho_n \rangle$, then for each $i \in \{1, \ldots, n\}$, $0 < \rho_i \le 1$. (Recall: *A smaller rate means a faster computer.*) We posit a uniform communication fabric for all computers: The time to send a single packet either from the master C_0 to some worker C_i or from C_i to C_0 is τ time units.[4] Within the context of the CEP, every intercomputer message is either a work-allocation that C_0 sends to some $C_i \in \mathcal{C}$ or the results of executed work that C_i sends to C_0. We posit that *each unit of work produces* $0 < \delta < 1$ *units of results*. The entire L-time-unit "exploitation" episode must be orchestrated so that *at most one intercomputer message is in transit in the network at a time*. Before any C_i sends a message of length ℓ to another C_j, C_i *packages* the message, at a cost of $\pi_i \ell$ time-units; symmetrically, when C_j receives the message, it *unpackages* it, at a cost of $\overline{\pi}_i \ell$ time-units. (Packaging a message could be as computationally "lightweight" as packetizing and compressing it or as "heavyweight" as encoding it.)

Worksharing protocols. When there is only *one* $C_i \in \mathcal{C}$, C_0 shares w units of work with C_i via the process summarized in the following schematic time-line of worksharing with one worker computer (not to scale).

C_0 packages work for C_i	work is in transit	C_i unpackages the work	C_i computes the work	C_i packages its results	results are in transit	C_0 unpackages the results
$\pi_0 w$	τw	$\overline{\pi}_i w$	$\rho_i w$	$\pi_i \delta w$	$\tau \delta w$	$\overline{\pi}_0 \delta w$

When there are *many* (i.e., more than one) $C_i \in \mathcal{C}$, we use two ordinal-indexing schemes for $\mathcal{C}$'s computers to help orchestrate communications while solving the CEP. The *startup order* specifies the order in which C_0 transmits work within $\mathcal{C}$; it labels the workers $C_{s_1}, \ldots, C_{s_n}$, to indicate that C_{s_i} receives work—hence, begins working—before $C_{s_{i+1}}$. Dually, the *finishing order* labels the workers $C_{f_1}, \ldots, C_{f_n}$, to specify the order in which they return their results to C_0. Protocols proceed as follows.

1. *Transmit work.* C_0 prepares and transmits w_{s_1} units of work for C_{s_1}. It immediately prepares and sends w_{s_2} units of work to C_{s_2} via the same process. Continuing thus, C_0 supplies the C_{s_i} with w_{s_i} units of work seriatim—with no intervening gaps.
2. *Compute.* Upon receiving work from C_0, C_i unpackages and performs the work.
3. *Transmit results.* As soon as C_i completes its work, it packages its results and transmits them to C_0.

[4] We find the transit rate τ a more convenient cost measure than its reciprocal, bandwidth.

We choose the work-allocations $\{w_i\}_{i=1}^n$ so that, with no gaps, $\mathcal{C}$'s computers:

– receive work and compute in the startup order $\Sigma = \langle s_1, \ldots, s_n \rangle$;
– complete work and transmit results in the finishing order $\Phi = \langle f_1, \ldots, f_n \rangle$;
– complete all work and communications by time L.

Our goal is to maximize $\mathcal{C}$'s *aggregate completed work*, $\mathcal{W}(\mathcal{C}; L) \overset{\text{def}}{=} w_1 + \cdots + w_n$.

In depicting and analyzing multiworker protocols, we have all computing by the master C_0—i.e., its packaging work-allocations for $\mathcal{C}$'s workers and unpackaging their results—take place *offline*, so that we focus solely on a worksharing episode as it appears to the workers. Although this choice differs from that in [2], one verifies easily that *all qualitative conclusions in [2]*—notably the optimality of FIFO protocols (cited as our Theorem 1)—*are independent of this choice*.

The timelines for two instantiations of the generic multiworker worksharing protocol appear in Fig. 1. (*To save space while preserving legibility we have each* $s_i \equiv i$.) In the top protocol, Σ and Φ *coincide:* $(\forall i)[f_i = s_i]$, which specifies the (optimal) *FIFO* protocol. In the bottom one, Σ and Φ *are reversed:* $(\forall i)[f_i = s_{n-i+1}]$, which specifies the *LIFO* protocol. Neither relationship is true of general protocols; cf. [2].

The following abbreviations enhance the legibility of complicated expressions.

– For $1 \le i \le n$, $\mathsf{R}_i \overset{\text{def}}{=} \overline{\pi}_i + \rho_i + \delta\pi_i$
  is the *effective computing rate* of computer C_i per work unit, i.e., the "round-trip" time-cost for [work-unpackaging + work-performing + result-packaging]
– $\widetilde{\tau} \overset{\text{def}}{=} (1+\delta)\tau$
  is the (common) per-unit "round-trip" communication rate for each computer.

Henceforth, we focus on a cluster $\mathcal{C}$ *with* effective *(heterogeneity) profile* $\langle \mathsf{R}_1, \ldots, \mathsf{R}_n \rangle$.

C_0 sends : work $\longrightarrow C_1$	work $\longrightarrow C_2$	work $\longrightarrow C_3$				
τw_1	τw_2	τw_3				
C_1 : waits	processes work			results $\longrightarrow C_0$		
IDLE	unpackage work $\overline{\pi}_1 w_1$	compute $\rho_1 w_1$	package results $\pi_1 \delta w_1$	send $\tau\delta w_1$	**IDLE**	**IDLE**
C_2 : waits	waits	processes work			results $\longrightarrow C_0$	
IDLE	**IDLE**	unpackage work $\overline{\pi}_2 w_2$	compute $\rho_2 w_2$	package results $\pi_2 \delta w_2$	send $\tau\delta w_2$	**IDLE**
C_3 : waits	waits	waits	processes work			results $\longrightarrow C_0$
IDLE	**IDLE**	**IDLE**	unpackage work $\overline{\pi}_3 w_3$	compute $\rho_3 w_3$	package results $\pi_3 \delta w_3$	send $\tau\delta w_3$

C_0 sends : work $\longrightarrow C_1$	work $\longrightarrow C_2$	work $\longrightarrow C_3$				
τw_1	τw_2	τw_3				
C_1 : waits	processes work					results $\longrightarrow C_0$
IDLE	unpackage work $\overline{\pi}_1 w_1$	compute $\rho_1 w_1$	package results $\pi_1 \delta w_1$			send $\delta\tau w_1$
C_2 : waits	waits	processes work			results $\longrightarrow C_0$	
IDLE	**IDLE**	unpackage work $\overline{\pi}_2 w_2$	compute $\rho_2 w_2$	package results $\pi_2 \delta w_2$	send $\tau\delta w_2$	**IDLE**
C_3 : waits	waits	waits	processes work	results $\longrightarrow C_0$		
IDLE	**IDLE**	**IDLE**	unpackage $\overline{\pi}_3 w_3$ compute $\rho_3 w_3$ package $\pi_3 \delta w_3$	send $\tau\delta w_3$	**IDLE**	**IDLE**

Fig. 1. Time-lines of 3-worker FIFO (top) and LIFO (bottom) protocols (not to scale)

3 Work Production under the LIFO and FIFO Protocols

We invoke the observation from [2] that the work production of any n-computer cluster $\mathcal{C}$ under any worksharing protocol can be calculated by solving a system of n linear equations in n unknowns (the w_i). Assuming that $\mathcal{C}$'s computers are served in the startup order $s_1, \ldots, s_n$, the asymptotic[5] work-allocations to its computers under a given worksharing protocol P, denoted $w_{s_1}^{(\mathsf{P})}, \ldots, w_{s_n}^{(\mathsf{P})}$, are specified by the following system.

$$\mathsf{C}^{(\mathsf{P})} \cdot \begin{pmatrix} w_{s_1}^{(\mathsf{P})} \\ w_{s_2}^{(\mathsf{P})} \\ \vdots \\ w_{s_n}^{(\mathsf{P})} \end{pmatrix} = \begin{pmatrix} L \\ L \\ \vdots \\ L \end{pmatrix}; \tag{1}$$

$\mathsf{C}^{(\mathsf{P})}$ is the *coefficient matrix* that specifies the details of protocol P. One can "read off" the coefficient matrices for the LIFO protocol L and the FIFO protocol F from Fig. 1:

$$\mathsf{C}^{(\mathsf{L})} = \begin{pmatrix} \mathsf{R}_{s_1} + \widetilde{\tau} & 0 & \cdots & 0 \\ \widetilde{\tau} & \mathsf{R}_{s_2} + \widetilde{\tau} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \widetilde{\tau} & \widetilde{\tau} & \cdots & \mathsf{R}_{s_n} + \widetilde{\tau} \end{pmatrix}; \quad \mathsf{C}^{(\mathsf{F})} = \begin{pmatrix} \mathsf{R}_{s_1} + \widetilde{\tau} & \delta\tau & \cdots & \delta\tau \\ \tau & \mathsf{R}_{s_2} + \widetilde{\tau} & \cdots & \delta\tau \\ \vdots & \vdots & \ddots & \vdots \\ \tau & \tau & \cdots & \mathsf{R}_{s_n} + \widetilde{\tau} \end{pmatrix} \tag{2}$$

One can solve system (1) instantiated with the coefficients from (2) to determine the work-allocations $\{w_{s_i}^{(\mathsf{L})}\}_{i=1}^n$ and $\{w_{s_i}^{(\mathsf{F})}\}_{i=1}^n$, hence also the resulting amounts of aggregate completed work, $\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L) = \sum_i w_{s_i}^{(\mathsf{L})}$ and $\mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L) = \sum_i w_{s_i}^{(\mathsf{F})}$.

Theorem 1 ([2]). (a) *For any cluster $\mathcal{C}$ and lifespan L,*

$$\mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L) = \frac{X}{1 + \delta\tau X} \cdot L \quad \textit{where} \quad X = \sum_{k=1}^{n} \frac{1}{\mathsf{R}_{s_k} + \tau} \cdot \prod_{i=1}^{k-1} \frac{\mathsf{R}_{s_i} + \delta\tau}{\mathsf{R}_{s_i} + \tau}. \tag{3}$$

(b) *No worksharing protocol has greater work production than the FIFO protocol.*

Theorem 2. *For any cluster $\mathcal{C}$ and lifespan L,*

$$\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L) = L \cdot \sum_{k=1}^{n} \frac{1}{\mathsf{R}_{s_k} + \widetilde{\tau}} \cdot \prod_{i=1}^{k-1} \frac{\mathsf{R}_{s_i}}{\mathsf{R}_{s_i} + \widetilde{\tau}}. \tag{4}$$

Proof (Sketch). By considering the equation for $w_{s_1}^{(\mathsf{L})}$, plus all pairs of adjacent equations (i.e., equations whose indices have the forms s_i and s_{i+1}), we find that

$$\left[w_{s_1}^{(\mathsf{L})} = \frac{1}{\mathsf{R}_{s_1} + \widetilde{\tau}} \cdot L \right] \text{ and } \left[w_{s_k}^{(\mathsf{L})} = \frac{\mathsf{R}_{s_{k-1}}}{\mathsf{R}_{s_k} + \widetilde{\tau}} \cdot w_{s_{k-1}}^{(\mathsf{L})} \text{ for } k \in \{2, \ldots, n\} \right] \tag{5}$$

[5] Throughout, *asymptotic* means "as L grows without bound."

By unfolding the recurrent portion of (5), we find explicit expressions for each $w_{s_k}^{(\mathsf{L})}$:

$$w_{s_k}^{(\mathsf{L})} = \frac{1}{\mathsf{R}_{s_k} + \widetilde{\tau}} \cdot \prod_{i=1}^{k-1} \frac{\mathsf{R}_{s_i}}{\mathsf{R}_{s_i} + \widetilde{\tau}} \cdot L. \qquad \square$$

Theorems 1(a) and 2 specify the work productions of (respectively) the FIFO and LIFO protocols for a given, but unspecified, startup order $s_1, \ldots, s_n$. The fact that the notations $\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L)$ and $\mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L)$ do not specify this order presages the fact that these quantities are, in fact *independent of startup order.*

Theorem 3 ([2,4]). *When cluster $\mathcal{C}$ is scheduled according to either the LIFO protocol or the FIFO protocol, its aggregate completed work is independent of the order in which $\mathcal{C}$'s computers are served. That is, for all startup orders Σ_1 and Σ_2:*

$$\left[\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L; \Sigma_1) = \mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L; \Sigma_2)\right] \quad \textit{and} \quad \left[\mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L; \Sigma_1) = \mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L; \Sigma_2)\right].$$

This result appears in [2] for the FIFO protocol and can be derived from results in [4] for the LIFO protocol. We present an alternative proof for the LIFO protocol by emulating the elegant proof strategy used in [12] for the FIFO protocol.

A function $F(x_1, \ldots, x_n)$ is symmetric if its value is unchanged by every reordering of values for its variables. When $n = 3$, for instance, we must have

$$F(a, b, c) = F(a, c, b) = F(b, a, c) = F(b, c, a) = F(c, a, b) = F(c, b, a)$$

for all values a, b, c for the variables x_1, x_2, x_3. For integers $n > 1$ and $k \in \{1, \ldots n\}$, $F_k^{(n)}$ denotes the *multilinear* symmetric function[6] that has n variables grouped as products of k variables. The four functions $F_k^{(4)}$ of R-values appear in the following table.

$F_1^{(4)}(\mathsf{R}_1, \mathsf{R}_2, \mathsf{R}_3, \mathsf{R}_4)$	$= \mathsf{R}_1 + \mathsf{R}_2 + \mathsf{R}_3 + \mathsf{R}_4$
$F_2^{(4)}(\mathsf{R}_1, \mathsf{R}_2, \mathsf{R}_3, \mathsf{R}_4)$	$= \mathsf{R}_1\mathsf{R}_2 + \mathsf{R}_1\mathsf{R}_3 + \mathsf{R}_1\mathsf{R}_4 + \mathsf{R}_2\mathsf{R}_3 + \mathsf{R}_2\mathsf{R}_4 + \mathsf{R}_3\mathsf{R}_4$
$F_3^{(4)}(\mathsf{R}_1, \mathsf{R}_2, \mathsf{R}_3, \mathsf{R}_4)$	$= \mathsf{R}_1\mathsf{R}_2\mathsf{R}_3 + \mathsf{R}_1\mathsf{R}_2\mathsf{R}_4 + \mathsf{R}_1\mathsf{R}_3\mathsf{R}_4 + \mathsf{R}_2\mathsf{R}_3\mathsf{R}_4$
$F_4^{(4)}(\mathsf{R}_1, \mathsf{R}_2, \mathsf{R}_3, \mathsf{R}_4)$	$= \mathsf{R}_1\mathsf{R}_2\mathsf{R}_3\mathsf{R}_4$

Two notational simplifications will enhance legibility. (1) We allow k to assume the value 0 and set $F_0^{(n)} \equiv 1$. (2) Because our arguments to the functions $F_i^{(n)}$ are always $\mathsf{R}_1, \ldots, \mathsf{R}_n$, we abbreviate "$F_i^{(n)}(\mathsf{R}_1, \ldots, \mathsf{R}_n)$" by "$F_i^{(n)}$."

Theorem 3 is immediate from the following lemma.

Lemma 1. *For all lifespans L:*

$$\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L) = L \cdot \frac{F_{n-1}^{(n)} + F_{n-2}^{(n)}\widetilde{\tau} + \cdots + F_1^{(n)}\widetilde{\tau}^{n-2} + F_0^{(n)}\widetilde{\tau}^{n-1}}{(\mathsf{R}_1 + \widetilde{\tau}) \times (\mathsf{R}_2 + \widetilde{\tau}) \times \cdots \times (\mathsf{R}_{n-1} + \widetilde{\tau}) \times (\mathsf{R}_n + \widetilde{\tau}).} \qquad (6)$$

Thus, $\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L)$ is symmetric *in the effective computing rates,* $\mathsf{R}_1, \ldots, \mathsf{R}_n$.

Proof. We proceed by induction on the sizes (i.e., numbers of computers) of clusters. To aid legibility, we embellish the "name" of each cluster $\mathcal{C}$ with a subscript that specifies

[6] The qualifier "multilinear" tells us that no variable occurs to a power > 1; this excludes symmetric functions such as $F(a, b) = a^2b + ab^2$.

its size. The notational convention is that the n-computer cluster $\mathcal{C}_n$ has effective profile $\langle \mathsf{R}_1, \ldots, \mathsf{R}_n \rangle$. As the base of our induction, we note from equation (4) that

$$\mathcal{W}^{(\mathsf{L})}(\mathcal{C}_1; L) = \frac{1}{(\mathsf{R}_1 + \widetilde{\tau})} = \frac{F_0^{(1)}}{(\mathsf{R}_1 + \widetilde{\tau})};$$

$$\mathcal{W}^{(\mathsf{L})}(\mathcal{C}_2; L) = \frac{1}{(\mathsf{R}_1 + \widetilde{\tau})} + \frac{1}{(\mathsf{R}_2 + \widetilde{\tau})} \cdot \frac{\mathsf{R}_1}{(\mathsf{R}_1 + \widetilde{\tau})} = \frac{F_1^{(2)} + F_0^{(2)}\widetilde{\tau}}{(\mathsf{R}_1 + \widetilde{\tau}) \times (\mathsf{R}_2 + \widetilde{\tau})}.$$

Now assume, for induction, that (6) holds for all cluster sizes up through n. Combining our two specifications of $\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L)$, viz., equations (6) and (4), we then have

$$\begin{aligned}\mathcal{W}^{(\mathsf{L})}(\mathcal{C}_{n+1}; L) &= \mathcal{W}^{(\mathsf{L})}(\mathcal{C}_n; L) + \frac{1}{\mathsf{R}_{n+1} + \widetilde{\tau}} \cdot \prod_{i=1}^{n} \frac{\mathsf{R}_i}{\mathsf{R}_i + \widetilde{\tau}} \\ &= \frac{\left(F_{n-1}^{(n)} + F_{n-2}^{(n)}\widetilde{\tau} + \cdots + F_0^{(n)}\widetilde{\tau}^{n-1}\right)(\mathsf{R}_{n+1} + \widetilde{\tau}) + \prod_{i=1}^{n} \mathsf{R}_i}{(\mathsf{R}_1 + \widetilde{\tau}) \times (\mathsf{R}_2 + \widetilde{\tau}) \times \cdots \times (\mathsf{R}_n + \widetilde{\tau}) \times (\mathsf{R}_{n+1} + \widetilde{\tau})}.\end{aligned}$$

The proof is now completed by invoking the following easily verified identities:

$$\text{For all } i \in \{1, \ldots, n\}, \quad \mathsf{R}_{n+1} F_{n-i}^{(n)} + F_{n-i+1}^{(n)} = F_{n-i+1}^{(n+1)}.$$

In verifying these identities, recall that $\prod_{i=1}^{n} \mathsf{R}_i = F_n^{(n)}$. In applying these identities, note the role of the $\widetilde{\tau}$ in the induction-extending factor $(\mathsf{R}_{n+1} + \widetilde{\tau})$. □

A conjecture. Olivier Beaumont has reported, in personal communication, that informal simulations have failed to turn up any other worksharing protocol that shares the LIFO and FIFO protocols' independence from the order in which cluster $\mathcal{C}$'s computers are supplied with work. It is intriguing to conjecture that these two protocols are, in fact, unique in this independence. A plausible place to start trying to prove this is to exploit the fact that the LIFO and FIFO protocols are the only ones whose coefficient matrices—cf. (2)—have upper triangles and lower triangles that are all constant. (This fact about the matrices can be verified from the development in [2].)

Theorem 3 combines with equations (3) and (4) to reveal the following simple but consequential facts. The first fact is that *LIFO and FIFO protocols complete more work on faster clusters.*

Proposition 1. *Let clusters $\mathcal{C}$ and $\mathcal{C}'$ have respective profiles $\langle \mathsf{R}_1, \ldots, \mathsf{R}_i, \ldots, \mathsf{R}_n \rangle$ and $\langle \mathsf{R}_1, \ldots, \mathsf{R}'_i, \ldots, \mathsf{R}_n \rangle$. If $\mathsf{R}'_i < \mathsf{R}_i$,[7] then for all L:*

$$\left[\mathcal{W}^{(\mathsf{L})}(\mathcal{C}'; L) > \mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L)\right] \quad \textit{and} \quad \left[\mathcal{W}^{(\mathsf{F})}(\mathcal{C}'; L) > \mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L)\right].$$

Proof. The proof for the FIFO protocol appears in [12], so we focus on the LIFO protocol. We refine equation (4) to specifies the startup order Σ. (We actually already do this in the righthand expression in (4).) We choose any startup order Σ for $\mathcal{C}$, for which $s_n = i$; i.e., Σ has the form $\Sigma = \langle s_1, \ldots, s_{n-1}, i \rangle$. We then form the versions of

[7] Recall: the indicated inequality means that $\mathcal{C}'$'s ith computer is *faster* than cluster $\mathcal{C}$'s.

equation (4) that use startup order Σ with both of the indicated profiles. To enhance perspicuity, we write these versions in a way that emphasizes the fact that $\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L; \Sigma)$ and $\mathcal{W}^{(\mathsf{L})}(\mathcal{C}'; L; \Sigma)$ differ only in their first terms.

$$\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L; \Sigma) = \left(\frac{1}{\mathsf{R}_{s_n} + \widetilde{\tau}} \cdot \prod_{i=1}^{n-1} \frac{\mathsf{R}_{s_i}}{\mathsf{R}_{s_i} + \widetilde{\tau}} + \sum_{k=1}^{n-1} \frac{1}{\mathsf{R}_{s_k} + \widetilde{\tau}} \cdot \prod_{i=1}^{k-1} \frac{\mathsf{R}_{s_i}}{\mathsf{R}_{s_i} + \widetilde{\tau}} \right) \cdot L$$

$$\mathcal{W}^{(\mathsf{L})}(\mathcal{C}'; L; \Sigma) = \left(\frac{1}{\mathsf{R}'_{s_n} + \widetilde{\tau}} \cdot \prod_{i=1}^{n-1} \frac{\mathsf{R}_{s_i}}{\mathsf{R}_{s_i} + \widetilde{\tau}} + \sum_{k=1}^{n-1} \frac{1}{\mathsf{R}_{s_k} + \widetilde{\tau}} \cdot \prod_{i=1}^{k-1} \frac{\mathsf{R}_{s_i}}{\mathsf{R}_{s_i} + \widetilde{\tau}} \right) \cdot L$$

Because $\mathsf{R}'_{s_n} < \mathsf{R}_{s_n}$, we thereby find that

$$\mathcal{W}^{(\mathsf{L})}(\mathcal{C}'; L; \Sigma) - \mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L; \Sigma) \; = \; L \cdot \left(\frac{1}{\mathsf{R}'_{s_n} + \widetilde{\tau}} - \frac{1}{\mathsf{R}_{s_n} + \widetilde{\tau}} \right) \cdot \prod_{i=1}^{n-1} \frac{\mathsf{R}_{s_i}}{\mathsf{R}_{s_i} + \widetilde{\tau}} \; > \; 0,$$

so that $\mathcal{W}^{(\mathsf{L})}(\mathcal{C}'; L; \Sigma) > \mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L; \Sigma)$. □

The second fact is that *adding an additional computer to cluster $\mathcal{C}$ increases $\mathcal{C}$'s aggregate completed work under the LIFO protocol.* (It is shown in [4], by example, that this need not be the case with the FIFO protocol.)

Proposition 2. *Let cluster $\mathcal{C}'$ be obtained by adding a $(n+1)$th computer to cluster $\mathcal{C}$. Then $\mathcal{W}^{(\mathsf{L})}(\mathcal{C}'; L) > \mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L)$.*

Proof. Let the new computer, C, have effective computing rate R. Let each of $\mathcal{C}$'s computers retain its starting index within $\mathcal{C}'$, and let us assign C startup index s_{n+1}, so that $\mathsf{R} = \mathsf{R}_{s_{n+1}}$. (This is just for convenience: Theorem 3 tells us that we could assign C any starting order without changing the result.) Invoking equation (4), we find that

$$\mathcal{W}^{(\mathsf{L})}(\mathcal{C}'; L) \; - \; \mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L) \; = \; \frac{1}{\mathsf{R}_{s_{n+1}} + \widetilde{\tau}} \cdot \prod_{i=1}^{n} \frac{\mathsf{R}_{s_i}}{\mathsf{R}_{s_i} + \widetilde{\tau}}.$$

This difference is positive because each factor in the product is. □

4 The LIFO Protocol Is Approximately Optimal

This section is devoted to proving that every cluster completes a fixed fraction as much work under the LIFO protocol as under the optimal FIFO protocol.

Theorem 4. *The LIFO protocol is "approximately" optimal, in the following sense. For every cluster $\mathcal{C}$, there exists a fixed constant $\varphi_{\mathcal{C}} > 0$ that does not depend on how heterogeneous cluster $\mathcal{C}$ is (as measured by the relative speeds of its fastest and slowest computers) such that $\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L) \; > \; \varphi_{\mathcal{C}} \cdot \mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L)$.*[8]

We prove Theorem 4 by computing a lower bound on $\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L)$ and an upper bound on $\mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L)$ and comparing the results. We focus on an n-computer cluster $\mathcal{C}$ whose effective heterogeneity profile is $\langle \mathsf{R}_1, \ldots, \mathsf{R}_n \rangle$, where

- $\mathsf{R}^{(\mathrm{slow})} \stackrel{\mathrm{def}}{=} \max\{\mathsf{R}_1, \ldots, \mathsf{R}_n\}$ is the effective rate of $\mathcal{C}$'s *slowest* computer;
- $\mathsf{R}^{(\mathrm{fast})} \stackrel{\mathrm{def}}{=} \min\{\mathsf{R}_1, \ldots, \mathsf{R}_n\}$ is the effective rate of $\mathcal{C}$'s *fastest* computer.

[8] Recall that $\mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L)$ is the most work that cluster $\mathcal{C}$ can complete in time L under *any* worksharing protocol (Theorem 1).

4.1 A Lower Bound on the LIFO Work Production $\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L)$

Lemma 2. *For every cluster $\mathcal{C}$ and lifespan L,* $\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L) \; > \; \dfrac{1}{\mathsf{R}^{(\mathrm{slow})} + \widetilde{\tau}} \cdot L.$

Proof. By combining equation (4) for $\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L)$ with the "order-independent" Theorem 3 and the "faster-clusters-are-better" Proposition 1, we find that

$$\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L) \; \geq \; \frac{1}{\mathsf{R}^{(\mathrm{slow})} + \widetilde{\tau}} \sum_{k=0}^{n-1} \left(\frac{\mathsf{R}^{(\mathrm{slow})}}{\mathsf{R}^{(\mathrm{slow})} + \widetilde{\tau}} \right)^k \cdot L \; = \; \frac{1}{\widetilde{\tau}} \cdot \left(1 \; - \; \left(\frac{\mathsf{R}^{(\mathrm{slow})}}{\mathsf{R}^{(\mathrm{slow})} + \widetilde{\tau}} \right)^n \right) \cdot L \tag{7}$$

We next invoke the "bigger-clusters-are-more-powerful" Proposition 2 to remark that

$$\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L) \; \geq \; \mathcal{W}^{(\mathsf{L})}(\mathcal{C}'; L), \tag{8}$$

where $\mathcal{C}'$ is the two-computer subcluster of $\mathcal{C}$ whose profile is $\langle \mathsf{R}_1, \mathsf{R}_2 \rangle$, and $\mathsf{R}^{(\mathrm{slow})} \in \{\mathsf{R}_1, \mathsf{R}_2\}$. We then combine inequalities (7) and (8) to see that

$$\begin{aligned}
\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L) \; \geq \; \mathcal{W}^{(\mathsf{L})}(\mathcal{C}'; L) &\geq \frac{1}{\widetilde{\tau}} \cdot \left(1 \; - \; \left(\frac{\mathsf{R}^{(\mathrm{slow})}}{\mathsf{R}^{(\mathrm{slow})} + \widetilde{\tau}} \right)^2 \right) \cdot L \\
&= \frac{1}{\widetilde{\tau}} \cdot \left(1 \; - \; \frac{\mathsf{R}^{(\mathrm{slow})}}{\mathsf{R}^{(\mathrm{slow})} + \widetilde{\tau}} \right) \cdot \left(1 \; + \; \frac{\mathsf{R}^{(\mathrm{slow})}}{\mathsf{R}^{(\mathrm{slow})} + \widetilde{\tau}} \right) \cdot L \\
&> \frac{1}{\widetilde{\tau}} \cdot \left(1 \; - \; \frac{\mathsf{R}^{(\mathrm{slow})}}{\mathsf{R}^{(\mathrm{slow})} + \widetilde{\tau}} \right) \cdot L \\
&= \frac{1}{\mathsf{R}^{(\mathrm{slow})} + \widetilde{\tau}} \cdot L.
\end{aligned}$$

The lemma follows. □

4.2 An Upper Bound on the FIFO Work Production $\mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L)$

Lemma 3. *For every cluster $\mathcal{C}$ and lifespan L,* $\mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L) \; < \; \dfrac{1}{\tau} \cdot L.$

Proof. We simplify the development by replacing cluster $\mathcal{C}$'s characterizing parameters by a composite parameter that bounds its *computation-to-communication complexity:* the ratio $\kappa_{\mathcal{C}} \stackrel{\mathrm{def}}{=} \mathsf{R}^{(\mathrm{fast})}/\tau$. By combining equation (3) for $\mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L)$ with the "faster-clusters-are-better" Proposition 1, we then find that

$$\begin{aligned}
\mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L) &\leq \frac{1}{\tau} \cdot \left(1 \; - \; \frac{(1-\delta)(\mathsf{R}^{(\mathrm{fast})} + \delta\tau)^n}{(\mathsf{R}^{(\mathrm{fast})} + \tau)^n \; - \; \delta \cdot (\mathsf{R}^{(\mathrm{fast})} + \delta\tau)^n} \right) \cdot L \\
&= \frac{1}{\tau} \cdot \left(1 \; - \; \frac{(1-\delta)(\kappa_{\mathcal{C}} + \delta)^n}{(\kappa_{\mathcal{C}} + 1)^n \; - \; \delta \cdot (\kappa_{\mathcal{C}} + \delta)^n} \right) \cdot L \\
&= \frac{1}{\tau} \cdot \left(1 \; - \; \frac{1-\delta}{\Big((\kappa_{\mathcal{C}} + 1)/(\kappa_{\mathcal{C}} + \delta) \Big)^{\kappa_{\mathcal{C}} + (n - \kappa_{\mathcal{C}})} - \delta} \right) \cdot L.
\end{aligned}$$

We now invoke the classical inequality

$$\left(1 + \frac{x}{m}\right)^{m} \le e^{x},$$

which holds for all real positive x and m, to observe that

$$\left(\frac{\kappa_{\mathcal{C}}+1}{\kappa_{\mathcal{C}}+\delta}\right)^{\kappa_{\mathcal{C}}} = \left(1+\frac{1-\delta}{\kappa_{\mathcal{C}}+\delta}\right)^{\kappa_{\mathcal{C}}} \le \left(1+\frac{1-\delta}{\kappa_{\mathcal{C}}+\delta}\right)^{\kappa_{\mathcal{C}}+\delta} \le e^{1-\delta}$$

This inequality combines with our assumption that $\delta < 1$ to allow us to extend the preceding chain of inequalities on $\mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L)$. We find that

$$\mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L) \le \frac{1}{\tau} \cdot \left(1 - \frac{1-\delta}{e^{(1-\delta)(n-\kappa_{\mathcal{C}})} - \delta}\right) \cdot L < \frac{1}{\tau} \cdot L.$$

The lemma follows. □

4.3 The LIFO-FIFO Bounding Ratio

Finally, we combine the bounds of Lemmas 2 and 3 to conclude that for all clusters $\mathcal{C}$ and lifespans L,

$$\mathcal{W}^{(\mathsf{L})}(\mathcal{C}; L) > \frac{\tau}{\mathsf{R}^{(\mathrm{slow})} + \widetilde{\tau}} \cdot \mathcal{W}^{(\mathsf{F})}(\mathcal{C}; L).$$

The fraction $\varphi_{\mathcal{C}} = \dfrac{\tau}{\mathsf{R}^{(\mathrm{slow})} + \widetilde{\tau}}$ thus satisfies Theorem 4. □

Clearly, the fraction $\varphi_{\mathcal{C}}$ does not depend on how heterogeneous cluster $\mathcal{C}$ is as measured by the relative speeds of its fastest and slowest computers—as exposed, say, by the size of the ratio $\mathsf{R}^{(\mathrm{slow})}/\mathsf{R}^{(\mathrm{fast})}$.

5 Conclusions

We have exposed unexpected properties of the LIFO worksharing protocol, the structurally attractive and intuitively compelling *greedy* solution to the Cluster Exploitation Problem (CEP). These properties involve both the structure of the LIFO protocol and its behavior, measured via its work-production in the CEP.

In terms of the LIFO protocol's behavior, our main result shows that the protocol's work-production when solving the CEP is at least a fixed constant fraction of optimal (Theorem 4). In view of the ease of specifying and analyzing the LIFO protocol, this result may promote interest in the protocol—and in pursuing analogous performance bounds for other as-yet unanalyzed scheduling heuristics.

In terms of the LIFO protocol's structure, we have shown that a cluster's work-production under the LIFO protocol is independent of the order in which the cluster's computers are supplied with work (Theorem 3). This independence is shared by the optimal FIFO worksharing protocol; we conjecture that it is shared by no protocols other than FIFO and LIFO. This unexpected result joins companions in [2,10,12] in reminding us of the subtlety of the phenomenon of heterogeneity in computing—even with respect to as simple a scheduling problem as the CEP.

Future work will attempt to settle the order-independence conjecture and will explore the CEP when work complexity is not linear in work size.

Acknowledgments. It is a pleasure to thank the anonymous referees for their careful reviews, and Olivier Beaumont and Sanjay Rajopadhye for stimulating conversations.

References

1. Adler, M., Gong, Y., Rosenberg, A.L.: Asymptotically Optimal Worksharing in HNOWs: How Long Is 'Sufficiently Long'? In: 36th Annual Simulation Symposium, pp. 39–46 (2003)
2. Adler, M., Gong, Y., Rosenberg, A.L.: On "Exploiting" Node-Heterogeneous Clusters Optimally. Theory of Computing Systems 42, 465–487 (2008)
3. Beaumont, O., Legrand, A., Robert, Y.: The Master-Slave Paradigm with Heterogeneous Computers. IEEE Transactions on Parallel and Distributed Systems 14, 897–908 (2003)
4. Beaumont, O., Marchal, L., Robert, Y.: Scheduling Divisible Loads with Return Messages on Heterogeneous Master-Worker Platforms. In: Bader, D.A., Parashar, M., Sridhar, V., Prasanna, V.K. (eds.) HiPC 2005. LNCS, vol. 3769, pp. 498–507. Springer, Heidelberg (2005)
5. Beaumont, O., Rosenberg, A.L.: Link-Heterogeneity vs. Node-Heterogeneity in Clusters. In: 17th International High-Performance Computing Conference (2010)
6. Bharadwaj, V., Ghose, D., Mani, V.: Optimal Sequencing and Arrangement in Distributed Single-Level Tree Networks. IEEE Transactions on Parallel and Distributed Systems 5, 968–976 (1994)
7. Bharadwaj, V., Ghose, D., Mani, V., Robertazzi, T.G.: Scheduling Divisible Loads in Parallel and Distributed Systems. J. Wiley & Sons, New York (1996)
8. Cappello, F., Fraigniaud, P., Mans, B., Rosenberg, A.L.: An Algorithmic Model for Heterogeneous Clusters: Rationale and Experience. International Journal of Foundations of Computer Science 16, 195–216 (2005)
9. Dutot, P.-F.: Complexity of Master-Slave Tasking on Heterogeneous Trees. European Journal of Operational Research 164, 690–695 (2005)
10. Rosenberg, A.L.: On Sharing Bags of Tasks in Heterogeneous Networks of Workstations: Greedier Is Not Better. In: 3rd IEEE International Conference on Cluster Computing, pp. 124–131 (2001)
11. Rosenberg, A.L.: Changing Challenges for Collaborative Algorithmics. In: Zomaya, A. (ed.) Handbook of Nature-Inspired and Innovative Computing: Integrating Classical Models with Emerging Technologies, pp. 1–44. Springer, New York (2006)
12. Rosenberg, A.L., Chiang, R.C.: Toward Understanding Heterogeneity in Computing. In: 24th IEEE International Parallel and Distributed Processing Symposium, IPDPS 2010 (2010)