Guidelines for Data-Parallel Cycle-Stealing in Networks of Workstations

(Extended Abstract)

Arnold L. Rosenberg*
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
rsnbrg@cs.umass.edu

Abstract

We derive guidelines for nearly optimally scheduling data-parallel computations within a draconian mode of cycle-stealing in NOWs. In this computing regimen, workstation A takes control of workstation B's processor whenever B is idle, with the promise of relinquishing control immediately *upon demand—thereby losing work in progress. The typically high communication overhead for supplying workstation B with work and receiving its results militates in favor of supplying B with large amounts of work at a time; the risk of losing work in progress when B is reclaimed militates in favor of supplying B with a succession of small bundles of work. The challenge is to balance these two pressures in a way that maximizes (some measure of) the amount of work accomplished. Our guidelines attempt to maximize the expected work accomplished by workstation B in an episode of cycle-stealing, assuming knowledge of the instantaneous probability of workstation B's being reclaimed. Our study is a step toward rendering prescriptive the descriptive study of cycle-stealing in [3].*

1. The Cycle-Stealing Problem

We derive guidelines for (almost) optimally scheduling data-parallel computations on "borrowed" workstations, within the model developed in [3]. The phenomenological study in that paper builds on the following rather draconian version of *cycle-stealing*—the use by one workstation of idle computing cycles of another. The owner of workstation A contracts to take control of workstation B whenever its owner is absent. When the owner of B reclaims that workstation, workstation A *immediately* relinquishes control of B, killing any active job(s)—thereby destroying all work since the last checkpoint.

> Such draconian "contracts" are inevitable, for instance, when a returning owner unplugs a laptop from a network; one encounters such contracts also at several institutions where cycle-stealing is supported.

Such a "contract" creates a tension between the following inherently conflicting aspects of cycle-stealing. On the one hand, since any work in progress on workstation B when it is reclaimed is lost, a cycle-stealer wants to break a cycle-stealing episode into many short *periods*, supplying small amounts of work to the borrowed workstation each time. On the other hand, since each of the inter-workstation communications that bracket every period in a cycle-stealing episode—to supply work to workstation B and to reclaim the results of that work—involves an expensive setup protocol, the cycle-stealer wants to break each cycle-stealing episode into a few long periods, supplying large amounts of work to workstation B each time. Clearly, the challenge in scheduling episodes of cycle-stealing is to balance these conflicting factors in a way that maximizes the productive output of the episode. The research we report on here resolves the preceding conflict by deriving scheduling guidelines that (approximately) maximize the expected work[1] accomplished within an episode of cycle-stealing, within the following setting. We focus on computations that are data-parallel, in that they consist of a massive number of independent repetitive tasks of known durations.

> Many scientific computations have this form.

We develop schedules assuming that we know the instantaneous probability of workstation B's being reclaimed and that the function yielding this information is "smooth."

> Although our results are stated as though we had exact knowledge of these probabilities, they extend easily to situations wherein this knowledge is approximate,

*This research was supported in part by NSF Grant CCR-97-10367.

[1] In a forthcoming sequel, we focus on (nearly) optimizing other measures of a cycle-stealing episode's work output.

say, garnered from trace data that exposes B's owner's computer usage patterns. Our assumption of functional "smoothness" is reasonable, since one would likely encapsulate even trace data by some "well-behaved" curve.

Our hope—and experience—is that the scheduling guidelines we derive narrow the search space for a truly optimal schedule to manageable proportions.

Section 2 presents the formal model under which we derive our scheduling guidelines in Section 3. We illustrate the application of the guidelines in a variety of scenarios in Section 4 and end with open problems in Section 5.

Other noteworthy studies of scheduling algorithms for NOWs, which differ from ours in focus or objectives, appear in [1, 2, 4, 5, 6]. Of these, only [2] deals with the present adversarial scenario of stealing cycles; its main contribution is a randomized strategy that, with high probability, steals cycles within a logarithmic factor of optimally. We do not list the many empirical studies of computation on NOWs whose main foci are on enabling systems or specific applications rather than on analyzed scheduling algorithms.

Remark. The model we study here has applications to "real-life" problems other than scheduling single episodes of cycle-stealing. One important example is scheduling saves in a fault-prone computing system, as studied in [7]. This problem admits an abstract formulation that is formally similar to our model for cycle-stealing. Our model differs in many details from that of [7], and our research methodology differs dramatically from that paper's, but it is clear that our results can be adapted to apply in that setting also.

2. Modeling Data-Parallel Cycle-Stealing

We review the basic structure of the cycle-stealing model of [3], focusing only on details that are relevant to the current study. We refer the reader to that paper for additional details and variations on the model presented here.

2.1. The Model

Overview. We schedule data-parallel cycle-stealing in an "architecture-independent" fashion, in the sense of [8]: the cost of inter-workstation communications is characterized by a single (overhead) parameter c, which is the (combined) cost of initiating both the communication in which workstation A sends work to workstation B, and the communication in which B returns the results of the work. We assume that: tasks are indivisible; task times may vary but are known perfectly; the time for a task includes the marginal cost of transmitting its input and output data (so we may keep c independent of the sizes of data transmissions).

Cycle-stealing schedules. Workstation A schedules an episode of cycle-stealing by partitioning the time of B's potential availability into a sequence of nonoverlapping *periods*. For simplicity, we identify a cycle-stealing schedule $\mathcal{S}$ with its sequence of period-lengths: $\mathcal{S} = t_0, t_1, \ldots$, where each $t_i > 0$. A schedule can be finite, when there is a known upper bound L on the length of the episode, or it can be infinite, when no such bound is known.[2] The intended interpretation is that at time

$$\tau_k \stackrel{\text{def}}{=} \begin{cases} 0 & \text{if } k = 0 \\ T_{k-1} \stackrel{\text{def}}{=} t_0 + t_1 + \cdots + t_{k-1} & \text{if } k > 0 \end{cases}$$

the kth period begins: workstation A supplies workstation B with an amount of work chosen so that t_k time units are sufficient for workstation A to send the work to workstation B, for workstation B to perform the work, and for workstation B to return the results of the work.

The work achieved by a schedule. The amount of work achievable in a period of length t_k is[3] $t_k \ominus c$. If workstation B *is not reclaimed* by time $T_k = \tau_k + t_k$, then the amount of work done so far during the episode is augmented by $t_k \ominus c$; if B *is reclaimed* by time T_k, then the episode ends, having accomplished work $\sum_{i=0}^{k-1} (t_i \ominus c)$. (This reckoning reflects the loss of work from the interrupted period.) Clearly, the risk of having a period interrupted, thereby losing work, may make it desirable to have the lengths of a bounded-lifespan schedule's *productive* periods (those with $t_i > c$) sum to *less than* the potential duration L of the episode.

Cycle-stealing with known risk. One cannot derive provably productive scheduling guidelines without some antidote for a malicious adversary who kills every episode of cycle-stealing just at the end of the 0th period. Here (as in the first half of [3]), this antidote resides in our assuming that we know the risk of being interrupted in the midst of a period, in the form of the *life function* p for the episode: for each time t, $p(t)$ is the probability that workstation B *has not been reclaimed by time* t. In accord with the motivating scenario: $p(0) = 1$; when an upper bound L to the duration of the episode is known, then p decreases monotonically to 0 in the range $0 \le t \le L$; when no bound L is known, then p decreases monotonically for all t, with the limit 0. In order to enable our analytical results, we let period-lengths be arbitrary real numbers, and we consider only life functions that are "well-behaved," in the sense of being *differentiable* and *having no flex points*. These idealizations make even our "definitive" results just guidelines.

Our goal is to maximize the *expected work* in an episode of cycle-stealing. For any schedule $\mathcal{S} = t_0, t_1, \ldots$ and life function p, this quantity is given by

$$E(\mathcal{S}; p) \stackrel{\text{def}}{=} \sum_{i \ge 0} (t_i \ominus c) p(T_i). \tag{1}$$

[2] Examples of both situations appear later.

[3] The operator "$\ominus$" denotes *positive subtraction* and is defined by: $x \ominus y \stackrel{\text{def}}{=} \max(0, x - y)$.

The summation in (1) has upper limit $m-1$ for an m-period schedule and ∞ for an infinite schedule (when no duration bound is known).

Cycle-stealing schedule $\mathcal{S}$ is *optimal for life function* p if it maximizes the expected production of work, $E(\mathcal{S}; p)$, over all schedules for p. The guidelines we derive emerge from exposing the structure of optimal schedules.

2.2. Two Useful Technical Results

Our first result permits us henceforth to use ordinary subtraction rather than positive subtraction in calculations involving (1); it slightly strengthens the analogous result in [3] but follows from the same proof.

Proposition 1 *Any schedule $\mathcal{S}$ for a life function p can be replaced by a schedule $\mathcal{S}'$ such that:*

- $E(\mathcal{S}'; p) \geq E(\mathcal{S}; p)$*;*
- *each period of $\mathcal{S}'$—save the last, if $\mathcal{S}'$ is finite—has length $> c$.*

Our main results presuppose "nice" structure in the life function p; all require that p be differentiable; some require that p enjoy one of the following nice "shapes."

> The life function p is *concave* (resp., *convex*) if its derivative is everywhere nonincreasing (resp., nondecreasing): for all ξ and $\eta > \xi$, we have $p'(\xi) \geq p'(\eta)$ (resp., $p'(\xi) \leq p'(\eta)$).

The three life functions studied in [3] illustrate these properties. The *geometrically increasing risk* life function with risk factor $a > 1$ and potential lifespan L, $p_{a,L}(t) \stackrel{\text{def}}{=} (a^L - a^t)/(a^L - 1)$, is concave. The *geometrically decreasing lifespan* life function with risk factor $a > 1$, $p_a(t) \stackrel{\text{def}}{=} a^{-t}$, is convex. The *uniform-risk* life function with potential lifespan L, $p_L(t) \stackrel{\text{def}}{=} 1 - t/L$, is both concave and convex.

Our second result is sometimes useful in sharpening the bounds given by our guidelines for concave life functions. It follows from showing that successive period-lengths of an optimal schedule for a concave life function must decrease by at least c, i.e., that each $t_{k+1} \leq t_k - c$.

Proposition 2 *Every episode of cycle-stealing that has a* concave *life function p has finite potential lifespan L. An optimal schedule for such a p has a finite number m of periods, which satisfies*

$$m \;<\; \left\lceil \sqrt{\frac{2L}{c} + \frac{1}{4}} + \frac{1}{2} \right\rceil. \tag{2}$$

3. Our Scheduling Guidelines

Our guidelines for scheduling episodes of cycle-stealing emerge from the bounds in the following theorem, which partially specify the period-lengths of a (nearly) optimal schedule.

Theorem 1 *Say that the schedule $\mathcal{S} = t_0, t_1, \ldots$ is optimal for the differentiable life function p.*
(a) *The period-lengths of $\mathcal{S}$ are given implicitly by the inductive system of equations: for each period-index $k \geq 0$,*[4]

$$p(T_k) \;=\; -\sum_{j \geq k} (t_j - c) p'(T_j). \tag{3}$$

In computationally more useful form: for each period-index $k > 0$,

$$p(T_k) \;=\; p(T_{k-1}) + (t_{k-1} - c) p'(T_{k-1}). \tag{4}$$

(b) *If the life function p is* convex, *then*

$$\sqrt{\frac{1}{4} - \frac{p(t_0)}{cp'(t_0)}} + \frac{1}{2} \;<\; \frac{t_0}{c} \;\leq\; 2\sqrt{\frac{1}{4} - \frac{p(t_0)}{cp'(t_0)}} + 1. \tag{5}$$

If the life function p is concave, *then*

$$\sqrt{\frac{1}{4} - \frac{p(t_0)}{cp'(t_0)}} + \frac{1}{2} \;<\; \frac{t_0}{c} \;\leq\; 2\sqrt{\frac{1}{4} - \frac{p(t_0)}{cp'(t_0/2)}} + 1. \tag{6}$$

Proof Hints. Theorem 1(a) follows from the fact that schedule $\mathcal{S}$, being optimal, accomplishes at least as much work as does any version of $\mathcal{S}$ that is "shifted," in the following sense.

> The $\langle k, -\delta \rangle$*-shift,* $\mathcal{S}^{\langle k,-\delta\rangle}$, of $\mathcal{S}$ and the $\langle k, +\delta \rangle$*-shift,* $\mathcal{S}^{\langle k,+\delta\rangle}$, of $\mathcal{S}$ are the schedules
>
> $$\begin{aligned} \mathcal{S}^{\langle k,-\delta\rangle} &\stackrel{\text{def}}{=} t_0, t_1, \ldots, t_{k-1}, t_k - \delta, t_{k+1}, \ldots, \\ \mathcal{S}^{\langle k,+\delta\rangle} &\stackrel{\text{def}}{=} t_0, t_1, \ldots, t_{k-1}, t_k + \delta, t_{k+1}, \ldots, \end{aligned}$$
>
> which have the same number of periods[5] as $\mathcal{S}$ and the same period-lengths, save for period k.

The lower bounds in Theorem 1(b) follow from the fact that an optimal schedule $\mathcal{S}$ is no less productive than the schedule $\widetilde{\mathcal{S}} \stackrel{\text{def}}{=} t_0 + t_1, t_2, \ldots$ that is obtained from $\mathcal{S}$ by combining the first two periods.

The upper bounds in Theorem 1(b) derive from the following lemma.

[4] The upper limit of each summation in (3) is inherited from (1).

[5] That is, if $\mathcal{S}$ has finitely many periods, then $\mathcal{S}^{\langle k,+\delta\rangle}$ and $\mathcal{S}^{\langle k,-\delta\rangle}$ have the same number; if $\mathcal{S}$ has infinitely many periods, then so also do $\mathcal{S}^{\langle k,+\delta\rangle}$ and $\mathcal{S}^{\langle k,-\delta\rangle}$.

Lemma 1 *Let $\mathcal{S} = t_0, t_1, \ldots$ be an optimal schedule for life function p. Either the initial period-length $t_0 \leq 2c$, or t_0 is small enough that*

$$p(t_0) \;\geq\; \max_{t \in (c,\; t_0 - c)} \left(1 - \frac{c}{t}\right) p(t). \tag{7}$$

The lemma follows from showing that any schedule $\mathcal{S} = t_0, t_1, \ldots$ that violates condition (7) can be improved (i.e., made more productive) by splitting its 0th period in two, yielding a schedule of the form $\widehat{\mathcal{S}} \stackrel{\text{def}}{=} \hat{t}, t_0 - \hat{t}, t_1, \ldots$.

One now derives the upper bounds in Theorem 1(b) by instantiating the system of inequalities that is implicit in (7) with the value $t = \frac{1}{2}t_0$, then invoking either the convexity or concavity of the life function p. ■

4. Applying the Guidelines

We illustrate the utility of the guidelines derived in the preceding section, by applying them to some specific life functions, producing for each life function an approximation to the optimal schedule $\mathcal{S} = t_0, t_1, \ldots$. Using system (4), we easily derive explicit expressions for each non-initial period-length t_k, where $k \geq 1$, in terms of all preceding period-lengths. Of course, these expressions are explicit only modulo our finding an explicit expression for t_0. We can only approximate this latter task, by using whichever pair of inequalities (5) or (6) is appropriate for the life function in question.

4.1. The Family $\{p_{d,L}(t) \stackrel{\text{def}}{=} 1 - t^d/L^d \mid d = 1, 2, \ldots\}$

We begin studying a family of concave life functions for an episode of cycle-stealing with potential lifespan L; the $(d = 1)$-member of the family is the life function for the *uniform risk scenario* of [3], wherein the risk of interruption is stable across the opportunity.

The non-initial period-lengths. The kth period-length t_k of an optimal schedule $\mathcal{S} = t_0, t_1, \ldots$ for $p_{d,L}$ can be determined as follows. Using system (4), we deduce that

$$t_k \;=\; \left(\left(1 + \frac{d(t_{k-1} - c)}{T_{k-1}}\right)^{1/d} - 1\right) T_{k-1}.$$

When $d = 1$, this expression simplifies even further, to

$$t_k \;=\; t_{k-1} - c, \tag{8}$$

as was discovered in [3].

The initial period-length. Since each $p_{d,L}$ is concave, we now invoke pair of inequalities (6) to obtain, after some manipulation, the following bounds on t_0.

$$\left(\frac{c}{d}\right)^{1/(d+1)} L^{d/(d+1)} \;\leq\; t_0 \;\leq\; 2\left(\frac{c}{d}\right)^{1/(d+1)} L^{d/(d+1)} + 1.$$

For specific values of d, we can use ad hoc techniques to get even tighter bounds. Most notably, when $d = 1$, we can revisit (3) and the proof of Proposition 2, in the light of (8) and the fact that $p'_{1,L} \equiv -1/L$, to deduce that t_0 is within additive constants of $\sqrt{2cL}$, as verified in [3].

4.2. The Family $\{p_a(t) \stackrel{\text{def}}{=} a^{-t} \mid a > e^{\frac{1}{2c}}\}$

The life functions in this family characterize the *geometrically decreasing lifespan* scenario of [3], which models a cycle-stealing opportunity that has a "half-life."

The non-initial period-lengths. Applying system (4) to p_a, we find that the non-initial period-lengths of an optimal schedule $\mathcal{S} = t_0, t_1, \ldots$ satisfy the recurrence

$$a^{-(T_{k-1} + t_k)} \;=\; a^{-T_{k-1}} - (t_{k-1} - c)a^{-T_{k-1}} \ln a,$$

so that

$$t_k \;=\; -\frac{1}{\ln a} \ln(1 - (t_{k-1} - c)\ln a). \tag{9}$$

Of course, system (9) can be solved for t_k only when each

$$t_i \;<\; \frac{1}{\ln a} + c.$$

The initial period-length. Since each p_a is convex, Lemma 1 and the pair of inequalities (5) combine to yield the following bounds on t_0.

$$\sqrt{\frac{c^2}{4} + \frac{c}{\ln a}} + \frac{c}{2} \;\leq\; t_0 \;\leq\; \frac{1}{\ln a} + c.$$

Not unexpectedly, an ad hoc analysis based on the specific value of a and the functional form of p_a yields tighter, albeit implicit, bounds; cf. [3].

4.3. The Family $\{p_a(t) \stackrel{\text{def}}{=} (a^L - a^t)/(a^L - 1) \mid a \geq 1\}$

The life functions in this family characterize the *geometrically increasing risk* scenario of [3], which models a cycle-stealing opportunity such as a coffee break, wherein the risk of interruption increases very fast.

The non-initial period-lengths. Applying system (4) to p_a, we find that the non-initial period-lengths of an optimal schedule $\mathcal{S} = t_0, t_1, \ldots$ satisfy the recurrence

$$a^L - a^{T_{k-1} + t_k} \;=\; a^L - a^{T_{k-1}} - (t_{k-1} - c)a^{T_{k-1}} \ln a,$$

so that

$$t_k \;=\; \ln(1 + (t_{k-1} - c)\ln a). \tag{10}$$

The initial period-length. Since each p_a is concave, we invoke the pair of inequalities (6) to obtain bounds on t_0.

Without writing out the long expressions that these inequalities yield, we note only that they show that, to within low-order additive terms (which involve c, t_0, and L),

$$t_0 \;=\; \frac{L}{\log_a^2 L}.$$

Our approximate specification of the period-lengths of $\mathcal{S}$ are more detailed than one finds in [3].

5. Conclusion

Our experience with specific life functions, as illustrated in Section 4, suggests that, despite its implicit nature, system (4) easily determines each non-initial period-length of an optimal schedule in terms of all earlier period-lengths. Significantly, this "progressive" feature of the system allows one to determine t_{i+1} only after period i has ended. This means that, in principle, one could use *conditional*, rather than absolute, probabilities to determine schedule $\mathcal{S}$ progressively, period by period. Determining the initial period-length t_0 remains an art: System (4) does not help in this determination, and the $(k = 0)$-instance of system (3) is usually hard to apply, except in very special cases, such as the uniform risk scenario. The bounds on the optimal value of t_0 that we derive in Theorem 1 substantially narrow one's search space for the optimal t_0, at least for "smooth" life functions, but they usually still leave one with a factor-of-2 uncertainly in determining this value. Indeed, we view the primary open problem within the framework we have studied here to be the identification of broad classes of life functions for which one can determine the optimal initial period-length t_0 definitively. Although we succeeded in this determination for the three scenarios studied in [3], we did so only by using ad hoc techniques that were specific to each particular life function.

Even aside from determining t_0 definitively, much work remains to be done within the framework of our study. Most obviously, our results expose only *necessary* dependencies among optimal period-lengths; they do not demonstrate that using such period-lengths guarantees the (near) optimality of the resulting schedule. (We do show in the full paper that the dependencies do guarantee "local" optimality.) While avenues toward global optimality guarantees have eluded us, one possible approach would involve answering the following question. (Note the relevance of Theorem 1(a).)

> *Are optimal cycle-stealing schedules unique?*

Notably, each of the life functions studied in [3] admits a unique optimal schedule—but the techniques for verifying uniqueness there were specific to the individual life function. Yet another approach to guaranteeing optimality—at least for specific classes of life functions—would be to determine when specific scheduling recipes work. One natural such recipe is to choose period-lengths "greedily:" one would choose t_0 by maximizing the function $p_0(t) \stackrel{\text{def}}{=} (t - c)p(t)$, then choose t_1 by maximizing the function $p_1(t) \stackrel{\text{def}}{=} (t - c)p(t + t_0)$, and so on.

> *For what class of life functions is a "greedy" cycle-stealing schedule optimal? More generally, how close to optimal are "greedy" schedules?*

Easily, the "greedy" strategy yields the optimal schedule for the geometrically decreasing lifespan scenario. In quite another direction, we do not yet have an answer to even the following basic question.

> *For what class of life functions do there exist optimal cycle-stealing schedules?*

A final set of open questions involve more technical issues. Our current results demand smoothness and/or a nice "shape" in our life functions. Can these assumptions be weakened? In another direction: we have had to translate what is ideally a discrete problem into a continuous framework in order to derive our guidelines; this was true even in the case study of [3]. Can one show that our continuous guidelines yield valuable discrete analogues?

It is clear from this brief list of questions that many challenges remain in this important area of research.

References

[1] M.J. Atallah, C.L. Black, D.C. Marinescu, H.J. Siegel, and T.L. Casavant. Models and algorithms for coscheduling compute-intensive tasks on a network of workstations. *J. Parallel Distr. Comput.*, 16:319–327, 1992.

[2] B. Awerbuch, Y. Azar, A. Fiat, and F.T. Leighton. Making commitments in the face of uncertainty: How to pick a winner almost every time. *28th ACM Symp. on Theory of Computing*, pages 519–530, 1996.

[3] S.N. Bhatt, F.R.K. Chung, F.T. Leighton, and A.L. Rosenberg. On optimal strategies for cycle-stealing in networks of workstations. *IEEE Trans. Comp.*, 46:545–557, 1997.

[4] R. Blumofe and C.E. Leiserson. Space-efficient scheduling of multithreaded computations. *25th ACM Symp. on Theory of Computing*, pages 362–371, 1993.

[5] R. Blumofe and C.E. Leiserson. Scheduling multithreaded computations by work stealing. *35th IEEE Symp. on Foundations of Computer Science*, pages 356–368, 1994.

[6] R. Blumofe and D.S. Park. Scheduling large-scale parallel computations on networks of workstations. *3rd Intl. Symp. on High-Performance Distributed Computing*, pages 96–105, 1994.

[7] E.G. Coffman, Jr., L. Flatto, and A.Y. Krenin. Scheduling saves in fault-tolerant computations. *Acta. Inform.*, 30:409–423, 1993.

[8] C.H. Papadimitriou and M. Yannakakis. Towards an architecture-independent analysis of parallel algorithms. *SIAM J. Comput.*, 19:322–328, 1990.