オペレーティングシステム・期末試験の解答例と解説

2010 年度 E・O クラス (2011 年 2 月 8 日・試験時間 90 分)

1. (a) 全スレッドが `while` 文に入る前の状態を初期状態とする．初期状態では $s.\mathrm{count} = k$, $s.\mathrm{waiting} = \phi$ である．(3) が不変式であることを以下のように帰納法によって示す．

- 初期状態においては $\#CS = 0$, $\#P(s) = 0$, $\#V(s) = 0$ なので，明らかに (3) は成立する．
- ある状態において (3) が成立しているとする（帰納法の仮定）．そこから遷移可能な各状態において (3) が成り立つことを示せばよい．ここで $\#CS$, $\#P(s)$ あるいは $\#V(s)$ が変化する可能性があるのは $P(s)$ と $V(s)$ の実行に関する遷移のみなので，それぞれの場合について調べてみる．
  - $P(s)$ を実行したとする．$P(s)$ を実行できるのは状態 P2 のみであるため，実行したスレッドが待ち状態にならない場合は CS に入る．以下の (P1) および (P2) により実行後も (3) は成立する．

    (P1) $s.\mathrm{count} = 0$ の場合: 実行したスレッドは待ち状態になってまだ CS には入らない．よって $\#CS$ は不変．定義より $\#P(s)$ も不変．

    (P2) $s.\mathrm{count} > 0$ の場合: $\#P(s)$ は 1 増加し，実行したスレッドは CS に入るので $\#CS$ も 1 増加する．
  - V(s) を実行したとする．$V(s)$ を実行できるのは状態 P4 のみであるため，実行したスレッドは CS を抜ける．以下の (V1) および (V2) により実行後も (3) が成立する．

    (V1) $s.\mathrm{waiting} = \phi$ の場合: $\#V(s)$ は 1 減少し，実行したスレッドは CS を抜けるので $\#CS$ も 1 減少する．

    (V2) $s.\mathrm{waiting} \neq \phi$ の場合: (V1) と同様に $\#V(s)$ と $\#CS$ はそれぞれ 1 減少する．このとき，$s.\mathrm{waiting}$ のスレッドが 1 つ選ばれて待ち状態を抜けて CS に入る．よって $\#CS$ は 1 増加し，定義より $\#P(s)$ も 1 増加する．

以上より (3) は不変式である．

(b) (1)(2) より

$$0 \leq k + \#V(s) - \#P(s)$$

であり，これを変形して

$$\#P(s) - \#V(s) \leq k$$

とできる．これに (3) を使えば

$$\#CS \leq k$$

を得る．

(c) 弱いセマフォとは，$s.\mathrm{waiting}$ から選ばれるスレッドについて何の仮定もないもの，つまりどのスレッドが選ばれるかが非決定的であるものをいう．このとき，いつまでたっても $s.\mathrm{waiting}$ から選ばれず，飢餓状態になるスレッドがあるような実行も可能である．スレッドの実行についての公平性 (fairness) は $s.\mathrm{waiting}$ からのスレッドの選び方についての公平性とは独立である．したがって，例えば問 1(a) で $N > 2$ の場合，スレッドの実行が公平であっても s が弱いセマフォのときは飢餓状態になるスレッドが生じる．

一方，強いセマフォとは，$s.\mathrm{waiting}$ から（完全に非決定的ではなく）何らかの方針に従ってスレッドを選ぶようなものである．例えば $s.\mathrm{waiting}$ を FIFO キューにすればいつまでたっても選ばれないスレッドは生じず，飢餓状態になるスレッドもなくなる．

参考 2007 年度とほぼ同じ問題．セマフォの定義および諸性質については講義資料 No.4 を参照のこと．

2. (a) 12 回

(b) 9 回

解説 LRU アルゴリズムでは，ページフレームに存在するページのうち，最後にアクセスされた時刻が最も小さい（古い）ものを犠牲ページとする．ページフレーム数 4 および 5 のときの LRU アルゴリズムによるページ置換の様子を表 1 および表 2 に示す．

LRU アルゴリズムを素直に実現しようとすると，ページをアクセスした時刻を毎回どこかに記録する必要が生じる．メモリアクセスに関してはそのオーバーヘッドは致命的であるため，通常は LRU を近似したアルゴリズムを用いる．

(c) 8 回

解説 最適アルゴリズムでは，ページフレームに存在するページのうち最後にアクセスされる予定のものを犠牲ページとする．ページフレーム数 4 のときの最適 (optimal) アルゴリズムによるページ置換の様子を表 3 に示す．

最適アルゴリズムは，他のアルゴリズムと比べると同じページ参照列についてページフォルトの数が最小になる．これを実現するためには将来のページ参照をあらかじめ知る必要があるが，一般にこれは決定可能ではない（停止問題に帰着できる）．

(d) 正しくない．例えば表 4 のように，ページフレーム数が 5 の場合について問の例と同じ参照列を試みると，ページフォルトは 12 回起っている．ページフレーム数が 4 の場合は 11 回であったので，これは「ページフレーム数を増やせばページフォルト数は減る（増えない）」という主張の反例になっている．

解説 この現象は Belady の異常 (Belady's anomaly) と呼ばれている [1].

参考 ページ置換アルゴリズムについては講義資料 No.8 を参照のこと．

3. write-back キャッシュでは速度向上のためにディスクへの書き込みが遅延される．したがってシステムがクラッシュした場合にキャッシュの内容がディスクに反映されないことがある．i-node ブロックがこのようなキャッシュ上に乗っている場合，クラッシュした際にファイルシステム全体の整合性が保たれなくなる危険性がある．

参考 2007 年度と同じ問題．ファイルシステムの構造については講義資料 No.9 を参照のこと．

参考文献

[1] L. A. Belady, R. A. Nelson & G. S. Shedler, An anomaly in space-time characteristics of certain programs running in a paging machine, *Communications of the ACM*, 12 (6), pp. 349–353, 1969.

表 1: ページフレーム数 4 のときの LRU アルゴリズムによるページ置換

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
ページフォルト	F	F	F	F	F	F	F	F	F				F	F	F
	0	0	0	0	1	2	3	4	0	1	2	5	0	1	2
物理ページ割当		1	1	1	2	3	4	0	1	2	5	0	1	2	3
(ページフレーム数=4)			2	2	3	4	0	1	2	5	0	1	2	3	4
				3	4	0	1	2	5	0	1	2	3	4	5

表 2: ページフレーム数 5 のときの LRU アルゴリズムによるページ置換

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
ページフォルト	F	F	F	F	F				F				F	F	F
	0	0	0	0	0	1	2	3	4	4	4	4	5	0	1
物理ページ割当		1	1	1	1	2	3	4	0	1	2	5	0	1	2
(ページフレーム数=5)			2	2	2	3	4	0	1	2	5	0	1	2	3
				3	3	4	0	1	2	5	0	1	2	3	4
					4	0	1	2	5	0	1	2	3	4	5

表 3: ページフレーム数 4 のときの最適アルゴリズムによるページ置換

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
ページフォルト	F	F	F	F	F				F				F	F	
	0	0	0	0	0	0	0	0	0	0	0	0	3	3	3
物理ページ割当		1	1	1	1	1	1	1	1	1	1	1	1	4	4
(ページフレーム数=4)			2	2	2	2	2	2	2	2	2	2	2	2	2
				3	4	4	4	4	5	5	5	5	5	5	5

表 4: ページフレーム数 5 のときの FIFO アルゴリズムによるページ置換

参照列	0	1	2	3	4	0	1	2	5	0	1	2	3	4	5
ページフォルト	F	F	F	F	F				F	F	F	F	F	F	F
	0	0	0	0	0	0	0	0	1	2	3	4	5	0	1
物理ページ割当		1	1	1	1	1	1	1	2	3	4	5	0	1	2
(ページフレーム数=5)			2	2	2	2	2	2	3	4	5	0	1	2	3
				3	3	3	3	3	4	5	0	1	2	3	4
					4	4	4	4	5	0	1	2	3	4	5