Additional Errata for Online Computation and Competitive Analysis

Dear Mister El-Yaniv and Mister Borodin, we have read your interesting book on online computation and competitive analysis and have compiled the following list of errors (at least we think that they are errors):

Chapter 1

Page 8 line -10: The reference in "In fact as we show in Theorem 2.1", should be replaced by Theorem 1.2.

Chapter 2

Page 24 line -11 (-12 counting the footnote): The statement "the value of $b(x)$ at any stage of the game is simply the number (mod 2) of accesses to x so far" should have been something like "the value of $b(x)$ at any stage of the game is simply the initial value of $b(x)$ plus the number (mod 2) of accesses to x so far."

Page 25 line 5-8: It is stated that "the proof will be complete once we prove that the following two conditions hold: (i) for each event i, $\mathrm{E}[a_i] \leq \frac{7}{4} \cdot \mathrm{OPT}_i$, where OPT_i is the cost incurred by OPT during the ith event; and (ii) Φ_{last} is bounded below." This however is insufficient to prove the theorem, because the $-\frac{3}{4}|\sigma|$ isn't accounted for.

Page 25 line 8: The condition "Φ_{last} is bounded below" should be "Φ_i is bounded below for all i" according to the definition on page 10. Another possible fix is to change the definition.

Page 25 line 23: "$\mathbf{E}[a_i] = \frac{1}{2}(2+1) \leq \frac{3}{2} \cdot \mathrm{OPT}_i$" should be $\mathbf{E}[a_i] \leq \frac{1}{2}(2+1) = \frac{3}{2} \cdot \mathrm{OPT}_i$. The $\leq$ instead of $=$ is because it is only in the worst case that $\mathbf{E}[a_i] = \frac{1}{2}(2+1)$, on the average it might be lower. The $=$ instead of $\leq$ is because OPT_i is exactly 1 since OPT makes one paid exchange.

Page 25 line 24-25: "The more demanding part of the proof concerns the case in which the ith event is an access to y (by either BIT or OPT)." This means that each event pertains only to one of the algorithms, but the proof assumes that they both act on each event (e.g. in the calculation of $\mathbf{E}[A]$ on page 26 the combined effect of BIT and OPT is taken into consideration). Inequality (2.1) also demonstrates the problem because if it is assumed that only one of the algorithms acts on event i, then OPT_i will always be zero when BIT acts, and since the expected amortized cost is positive the proof won't work. The problem can be fixed by changing the sentence in the bracket to "(by both BIT and OPT)."

Page 26 line 10: x should be changed to y in "Since either algorithm may move x forward", because it is the element y which is being accessed.

Page 26 line 11: x should be changed to y in "the items preceding x".

Page 26 line 15: "$(j = 1, 2, \ldots, j-1)$" should read "$(j = 1, 2, \ldots, k-1)$".

Page 26 line 15-16: The sentence "Let X_j [...] be a random variable giving the contribution of the inversion $\langle y, x_j \rangle$ if it is created." ought to be "Let X_j [...] be a random variable giving the contribution of the inversions $\langle y, x_j\rangle$ and $\langle x_j, y\rangle$ if they are created." The reason for this is that inversions of the second kind gives a contribution to A which is computed as a sum of the X_js, and six lines further down in the text $\langle x_j, y\rangle$ inversions are actually responsible for $X_j = 1$.

Page 26 line 22: "$X_i = 1$ for $k' < j \leq k-1$" should be changed to "$X_i = 1$ for $k' \leq j \leq k-1$" because y is also moved in front of $x_{k'}$.

Page 27 line 18: The expected cost of RMTF is $l(2l+k-2) = 2l^2 + lk - 2l$ instead of $2l^2 + 2lk - 2l$ because on average it is necessary to access each of the l elements twice while they are at the back of the list and $k-2$ times while they are at the back.

Chapter 3

Page 33 line -5, -4 (-9, -8 counting the footnote): The definition of demand paging is not consistent with the way it is used in the text. In it's current wording it allows any number of page evictions on a page fault. This means that FWF is a demand paging algorithm (contrary to what is stated on page 36).

Another definition of demand paging might be: Demand paging algorithms only evict pages when a page fault occurs and they never evict more than one page in connection with each page fault.

Page 39 line 18-19: Unless we assume that LFD already has the k pages that are requested first in the cache (without paying for it), it might make a page fault on the first request and on the k+1st. Now if $|\sigma| = k+1$ we have $LFD(\sigma) = 2$, this contradicts lemma 3.2 which states that "LFD(σ) $\leq \frac{|\sigma|}{k}$" which in this case is equal to $\frac{k+1}{k} < 2$ for $k > 1$. If we change the lemma to "LFD(σ) $\leq \left\lceil \frac{|\sigma|}{k} \right\rceil$" then it should be correct.

This alteration causes troubles in the proof of theorem 3.6, because we get: $\frac{\text{ALG}(\sigma)}{\text{OPT}(\sigma)} \geq \frac{|\sigma|}{\lceil \frac{|\sigma|}{k} \rceil}$, but if we assume that $\exists m \in \mathbb{Z}^+ : |\sigma| = mk$ then the ceiling can be removed and the desired result is obtained. Note that it is alright only to consider the special case where $\exists m \in \mathbb{Z}^+ : |\sigma| = mk$ since the theorem states a lower bound.

Page 40 line -4: It is stated that "$L(\sigma) \geq k$", this is not true in situations where there are only a few short phases and the last one is incomplete. This has implications for the proof of theorem 3.7 where the inequality on the first line of page 41 isn't generally true, but the small error this introduces can be "hidden" by the additive constant.

A possible correction could be something like: "Assuming that p is large the last phase can be ignored."

Chapter 4

Page 50 line 6: The definition of the algorithm MARK states that "initially, all the pages are marked". The consequence of this is that if the first requests in the request sequence are to pages already in cache, then the marking "rhythm" of MARK won't correspond to the k-phases of the request sequence. For example let $k = 2$, let the cache of MARK be $\{a, q\}$ and let the request sequence be the following:

$\sigma = a\, b \,|\, c\, d \,|\, e\, f.$

The k-phase partitioning of the request sequence is:

$\sigma = a\, b \,|\, c\, d \,|\, e\, f$.

MARK already has a in the cache, but not b therefore it will incur a page fault when b is requested and all pages are marked at this time. This means that the marking phases of MARK will be:

$\sigma = a \,|\, b\, c \,|\, d\, e \,|\, f$.

Here it is seen that the phases of MARK don't follow the k-phases.

This will make troubles in the proof of theorem 4.3, but they can be removed by making the definition state that all pages are unmarked from the beginning.

Chapter 6

Page 82 line 14: The product "$(|S^1| \cdot |S^2| \cdots |S^n|)$" should be "$(|S^1| \cdot |S^2| \cdots |S^{n-1}|)$, since player 1 has 1 decision node, player two has $|S^1|$ decision nodes, player 3 has $|S^1| \cdot |S^2|$ and so on.

Page 95 line -13, -14 (-14, -15 counting the foot note): The phrase "It is clear that upon processing σ, all permutation algorithms will end in configuration j" isn't strictly true. It is actually only in the worst case that this will happen. To see this consider the cache $\{1, 3, 4\}$ and the permutation $\pi = (1 \to 2 \to 3 \to 4 \to 1)$ used on the k-phase $3\,1\,2$ which belongs to $\phi_k(4)$. After serving this request sequence the cache of PERM_π will be $\{1, 2, 4\}$, since the only page fault which occurs is on the request of page 2, but that page fault will evict page 3, which means that PERM_π won't end in configuration 4.

Theorem 6.5 however is still correct since the actual value is no worse than if it actually happened that all permutation algorithms will end in configuration j.

Chapter 8

Page 109 line -11 (-13 counting the footnote): The reference to theorem 6.2 ought to be to corollary 6.3 instead.

Chapter 10

Page 153 line -4: The "$=$" should be changed to a "$>$" because the sum on the right side doesn't include the request r_n, which makes ALG incur a non-zero cost. (Note that the sum sums $n-1$ distances corresponding to $n-1$ requests, which is one less than the n requests in σ

Page 154 line -8: In "[...] that include the first request", first should be replaced by current or latest.

Page 160 line -13: "SC(γ)" should be "SC_γ".

Page 168 line -2 (-4 counting the footnote): "Similarly, $w'(Y) = w(Y - y + r)$" needs to have "$+d(r, y)$" added, for the same reasons that $+d(r, x)$ is added to $w'(X) = w(X - x + r)$.

Page 169 line 9,12: It is (indirectly) stated that $X_1 - x + r = X_{xr}$ which isn't true since $X_{xr} = X - x + r$ and X_1 is part of a partition of X.

Page 171 line 11: It is stated that $w'(B) + w(A) \leq w(B) + w(A)$ "trivially becomes an equality" "if $r \in B$". That isn't true since it would mean that $w(A) = w'(A)$. However it is true that the inequality is trivially true if $r \in B$ because w is a strictly nondecreasing function.

Chapter 12

Page 203 line -3 (-5 counting the footnotes): The equation "$\mathrm{OPT}(\sigma) = s \cdot \mathrm{OPT}(\sigma')$" should be "$s \cdot \mathrm{OPT}(\sigma) = \mathrm{OPT}(\sigma')$", since the loads in σ' is equal to loads of σ multiplied by s.

Page 208 line 2: The last V_i in the equations "$|V_i - V_{i+1}| = \frac{N}{2^i} = |V_i|$" should be changed to U_i ($|V_i|$ is actually $\frac{N}{2^{i-1}}$).

Page 208 line -12: Equation 12.1 in the definition of $\mathrm{SLOWFIT}_\Lambda$ should be changed from $i = \arg\min_k\{l_j(k) + r_{j+1}(k) \leq 2\Lambda\}$ to $i = \min\{k \in \{1, \dots, N\} : l_j(k) + r_{j+1}(k) \leq 2\Lambda\}$. The original definition means that i is either the value of k which minimizes boolean values (the results of the comparisons in the set) or which minimizes the sum $l_j(k)+r_{j+1}$. The first of these possibilities doesn't make sense and the second one aren't what we want.

Page 208 line -1: The n in "Since $f < n$" should be substituted with N (f is a machine index, n is the number of jobs and N is the number of machines).

Page 211 line 10: "lo" should be removed from "$\max_\sigma lo \max_e \frac{L_n(e)}{\mathrm{OPT}(\sigma)}$".

Page 211-213: In the definition of the algorithm $\mathrm{ROUTE\text{-}EXP}_\Lambda$ the only stated restriction on the parameter γ is that is has to be greater than zero. On page 212 theorem 12.7 states that $\mathrm{ROUTE\text{-}EXP}_\Lambda = O(\log m) \cdot \Lambda$, without making any further demands on the value of γ, but the proof of theorem 12.7 requires that $\gamma < 1$ (line 1 on page 213). Therefore the requirement $\gamma < 1$ must be either included in the definition of the algorithm or in the formulation of the theorem and of corollary 12.8.

Page 212 line -4 (-5 counting the footnote): An e should be added beneath the Σ on the right side of the equation.

Page 214 line 11: The number of nodes in G_k is $2TN + 3T - 1$ instead of the stated $2TN + 3T$. To see this note that there is only $2T - 1$ t_{i+1}^k nodes because i is defined to be bounded by $0 \leq i < 2T - 1$.

Page 215 line 3: $L_1(t) \geq L_2(t) \geq L_{q(t)}(t)$ should be $L_1(t) \geq L_2(t) \geq \cdots \geq L_{q(t)}(t)$

Page 215 line 13: The conclusion "$l \geq \sqrt{2N}(1+o(1))$" is based upon the fact that $\frac{l(l+1)}{2} \geq N$ — this means that $l^2 + l \geq 2N$ which only proves that $l \geq \sqrt{2N - l}$, which is strictly smaller than $\sqrt{2N}(1+o(1))$. The problem can be fixed by changing the plus to a minus.

Page 217 line 14: It isn't always true that all the machines in the sets of allowable machines of the jobs from the set S are hardworking at time $s_{t(j)}$. This can be seen by considering the fact that the machine m *became* hard-working at this time, which means that the other allowable machines also could have been non-hard-working at the time. The consequence of this is that h might not be smaller that $\sqrt{N}$.

If the job $r_{t(j)}$ is simply excluded from S its' load must be included in the final computations and this will give rise to a too high upper bound. The solution to the problem is to exclude $r_{t(j)}$ but include r_j, this leads to a substitution of l_j by $l_{t(j)}$ in lines -7 and -6 (-9 and -8 counting the foot note), which gives the desired result.

Page 217 line 15: In "all machines in M_k are hardworking since at least time $s_{t(k)}$" it isn't defined what $s_{t(k)}$ is, but if we replace it by $s_{t(j)}$ then everything works out alright.

Page 217 line 17: $s_{t(k)}$ could (should?) be changed to $s_{t(j)}$.

Page 217 line 17: In "Because there are at most $\sqrt{N}$ hard-working machines at time $s_{t(k)}$ [$s_{t(j)}$ - see above] and all machines in each M_k remain hardworking throughout the time interval $[s_{t(j)}, s_j)$, we have" the last part ("all machines in each M_k remain hardworking throughout the time interval $[s_{t(j)}, s_j)$") isn't necessarily true since the B(s)s might be increasing and some of the machines in one M_k might not be part of the following M_ks, whereby it would be possible for them to become non-hard-working.

However the limit on the number of machines in the union of the M_ks is still correct (if the above correction is applied), because all the machines in the M_ks were hardworking just after time $s_{t(j)}$.

Page 217 line -6 (-8 counting the footnote): $B_{t(j)}$ should be replaced by $B(t(j))$.

Chapter 14

Page 265 line -6: The reference to theorem 6.2 should instead be to corollary 6.3. Theorem 6.2 only states each mixed algorithm has a behavioral equivalent, but what we need is the knowledge that all randomized algorithms have an equivalent mixed algorithm.

Page 267 line -13,-10 (-18,-15 counting the foot notes): The algorithms RPP_i is defined for $i = 0, 1, \ldots, k-1$, but in the definition of EXPO they are used for $i = 1, \ldots, k$.

Chapter 15

Page 316 line -10 (-13 counting the foot notes): The principle of insufficient reason can choose both a_3 and a_5 though it is stated that it *will* select a_5. This can be seen by noting that the sum of the costs of the rows of a_3 and a_5 both are 21.

Website

Additional result relating to open question 11.1: It is stated that "Bartal, Chrobak and Larmore have shown that for k=2 servers on the continuous real line, there is a randomized algorithm which is 158/78-competitive (i.e. the first algorithm achieving a competitive ratio less than 2 competitive against an oblivious adversary for a space with more than 3 points)", but 158/78 is actually $2\frac{1}{39}$.

We would like to thank Joan Boyar, Kim Skak Larsen and Sanne Wøhlk for proof reading this list.

Jens Svalgaard Frederiksen (svalle@imada.sdu.dk)
Henning Martinussen (hma@imada.sdu.dk)
Morten Monrad Pedersen (mortenm@imada.sdu.dk)
1st of february 2001