Variable-Processor Cup Games

Alek Westover

March 13, 2020

Introduction. The cup game is a classic game in computer science that models work-scheduling. In the cup game a filler and an emptier take turns adding and removing water (i.e. work) to the cups. We investigate a variant of the vanilla multiprocessor cup game which we call the ***variable-processor cup game*** in which the filler is allowed to change the number of processors p (the amount of water that the filler can add and the number of cups from which the emptier can remove water. This is a natural extention of the vanilla multi-processor cup game to when the resources available are variable. Note that although the restriction that the filler and emptier's resources vary together may seem artificial, this is the only way to conduct the analysis; the rationale behind giving the emptier and filler equal resources in the classical vanilla multi-processor cup game is that this is the only way to achieve upper and lower bounds. The equivalent rational holds for the motivation of the variable-processor cup game. Analysis of this game does provide information about how real-world systems will behave.

Apriori the fact that the number of processors can vary offers neither the filler nor the emptier a clear advantage: lower values of p mean that the emptier is at more of a discretization disadvantage but also mean that the filler can anchor fewer cups. We hoped that the variable-processor cup game could be simulated in the vanilla multiprocessor cup game, because the extra ability given to the filler does not seem very strong. The new version of the cup game arose as we tried to get a bound of $\Omega(\log p)$ backlog in the multiprocessor game against an oblivious filler, which would combine with previous results to give us a lower bound that matches our upper bound: $O(\log\log n + \log p)$. In Proposition 2 we prove that there is an oblivious filling strategy in the variable-processor cup game on n cups that achieve backlgo $\Omega(\log n)$ as desired. [1]

However, we also show that attempts at simulating the variable-processor cup game are futile because the variable-processor cup game is–surprisingly–fundamentally different from the multi-processor cup game, and thus impossible to simulate. This follows as a corollary of an ***Amplification Lemma*** for both the adaptive and oblivious filler.

The following paragraphs follow the structure:

1. Proposition: Base case of inductive argument in corollary
2. Lemma: Amplification Lemma, allows for inductive step in inductive argument
3. Corollary: Repeatedly amplify the base case backlog to get very large backlog

We proceed with our results.

Adaptive Lowerbound.

Proposition 1. *There exists an adaptive filling strategy for the variable-processor cup game on n cups that achieves backlog $\Omega(\log n)$, where fill is relative to the average fill of the cups, with negative fill allowed.*

Proof. Let $h = \frac{1}{4}\log n/2$ be the desired fill. We call a cup ***overpowered*** if it has fill at least h. If there exists an overpowered cup the proposition is immediately satisfied, so we assume without loss of generality that there are no overpowered cups. Denote the fill of a cup i by $\text{fill}(i)$. Let the ***positive tilt*** of a cup i be $\text{tilt}_+(i) = \max(0, \text{fill}(i))$, and let the positive tilt of a set S of cups be $\sum_{i\in S}\text{tilt}_+(i)$. Let the ***mass*** of a set of cups S be $\sum_{i\in S}\text{fill}(i)$. Let A consist of the $n/2$ fullest cups, and B consist of the rest of the cups.

If no cups are overpowered, then $\text{tilt}_+(A\cup B) < h\cdot n$. Assume for sake of contradiction that there are more than $n/2$ cups i with $\text{fill}(i) \le -2h$. The mass of those cups would be less than $-hn$, but there isn't

[1] Note that we have $\Omega(\log n)$ in this proposition instead of $\Omega(\log p)$ because the filler can increase the number of processors, so it increases the number of processors to $n-1$ to start. A nearly identical construction could be used to show that backlog $\Omega(\log p_{\max})$ can be achieved, where the number of processors starts at $p_{\max}$ and the filler does not ever increase the number of processors. However, using $p_{\max} = n$ is natural in the variable-processor cup game, so we do not consider the game with the restriction that the filler can not increase the number of processors above some $p_{\max} < n$.

enough positive tilt to oppose this, a contradiction. Hence there are at most $n/2$ cups i with $\text{fill}(i) \leq -2h$.

We set the number of processors equal to 1 and play a single processor cup game on $n/2$ cups that have fill at least $-2h$ (which must exist) for $n/2-1$ steps. We initialize our "active set" to be A, noting that $\text{fill}(i) \geq -2h$ for all cups $i \in A$, and remove 1 cup from the active set at each step. At each step the filler distributes water equally among the cups in its active set. Then, the emptier will chose some cup to empty from. If this cup is in the active set the filler removes it from the active set. Otherwise, the filler chooses an arbitrary cup to remove from the active set.

After $n/2-1$ steps the active set will consist of a single cup. This cup's fill has increased by $1/(n/2) + 1/(n/2-1) + \cdots + 1/2 + 1/1 = H_{n/2} \geq \log n/2 = 4h$. Thus such a cup has fill at least $2h$ now, so the proposition is statisfied ☐

Lemma 1 (The Adaptive Amplification Lemma)**.** *Given an adaptive filling strategy for achieving fill $f(k)$ in a cup in the variable-processor cup game on k cups, there exists a strategy for achieving* ***amplified*** *fill*

$$f'(k) \geq \frac{1}{2}(f(k/2) + f(k/4) + f(k/8) + \cdots .$$

Proof. If, at any point in the process that will be described, backlog is greater than $f'(k)$, then the filler stops and the Lemma is satisfied as the desired backlog has been achieved. Thus we assume without loss of generality for the rest of the proof that no cup ever exceeds fill $f'(k)$ during the course of our algorithm. That is, we assume that we don't achieve the desired backlog until the end of our process.

The main idea of this analysis is as follows:

1. Using f repeatedly, achieve average fill at least $\frac{1}{2}f(n/2)$ in $n/2$ cups.
2. Halve the number of processors
3. Recurse on the $n/2$ cups with high average fill.

Let A the ***anchor set*** be initialized to consist of the $n/2$ fullest cups, and let B the ***non-anchor set*** be initialized to consist of the rest of the cups. Let $h_l = f(k/2^l)$; the filler will achieve a set of at least $n_l/2 = n/2^l$ cups with average fill at least $h_l/2$ on the l-th level of recursion.

On the l-th level of recursion we will repeatedly apply f to the non-anchor set until the anchor set has the desired average fill. If at any point in this process the non-anchor has average fill lower than $-h_l/2$, then anchor set has average fill at least $h_l/2$, so the process is finished. So long as B has average fill at least $-h_l/2$ then we would like to apply f to B. This would get fill $-h_l/2 + f(n_l/2) = h_l/2$ in some non-anchor cup. We replace the lowest cup in the anchor set with this cup.

A slight complication with this method is that we are anchoring the anchor set, and assuming that the emptier allways empties from each anchor cup; this may not be the case. However, the issue can be resolved by applying f up to $h_l n_l/4+1$ times per anchor cup.

Say that the emptier ***neglects*** the anchor set on an application of f if there is some step during the application of f in which the emptier does not empty from some anchor cup. Note that each time the emptier neflects the anchor set the mass of the anchor set increases by 1. If the emptier neglects the anchor set $h_l n_l/4 + 1$ times, then the average fill in the anchor set increases by more than $h_l/2$, so the desired fill is achieved in the anchor set.

Otherwise, there must have been an application of f for which the emptier did not neglect the anchor set. We only swap a cup into the anchor set if this is the case. In this case we actually do achieve fill $-h_l/2 + f(n_l/2)$ in a non-anchor cup, and swap it into the anchor set, as descibred before. ☐

Corollary 1. *There is an adaptive filling strategy for the variable-processor cup game on n cups that achieves backlog $\Omega(poly(n))$ in running time $2^{O(\log^2 n)}$*

Proof. Let

$$f_0(k) = \begin{cases} \log_2 k, & k \geq 1, \\ 0 & \text{else.} \end{cases}$$

Let f_{m+1} be the result of applying The Amplification Lemma to f_m. By repeated amplification $\log_2 n^{1/9}$ times we achieve a function $f_{\log_2 n^{1/9}}(k)$ with the property that for $k \geq n$, $f_{\log_2 n^{1/9}}(k) \geq 2^{\log_2 n^{1/9}} \log_2 k$. In particular, this gives a filling strategy that when applied to n cups gives backlog $\Omega(n^{1/9} \log_2 n) \geq \Omega(\text{poly}(n))$ as desired. To prove this, we prove the following lowerbound for f_m by induction:

$$f_m(k) \geq 2^m \log_2 k, \text{ for } k \geq (2^9)^m.$$

The base case follows from the definition of f_0. Assuming the property for f_m, we get the following:

for $k > (2^9)^{m+1}$,

$$\begin{aligned}
&f_{m+1}(k) \\
&= \frac{1}{2}(f_m(k/2) + f_m(k/4) + \cdots + f_m(k/2^9) + \cdots) \\
&\geq \frac{1}{2}(f_m(k/2) + f_m(k/4) + \cdots + f_m(k/2^9)) \\
&\geq \frac{1}{2}2^m(\log_2(k/2) + \log_2(k/4) + \cdots + \log_2(k/2^9)) \\
&\geq \frac{1}{2}2^m(9\log_2(k) - \frac{9 \cdot 10}{2}) \\
&\geq 2^{m+1}\log_2(k),
\end{aligned}$$

as desired. Hence the inductive claim holds, which establishes that $f_{\log_2 n^{1/9}}$ satisfies the desried condition, which proves that backlog can be made $\Omega(\text{poly}(n))$.

Remark 1. *The recursive construction requires quite a lot of steps, in fact a superpolynomial number of steps. If we consider the tree that represents computation of $f_{\log n^{1/\alpha}}(n)$ we see that each node will have at most α (some constant, e.g. $\alpha = 9$, α is the number of terms that we keep in the sum) children (the children of $f_k(c)$ are $f_{k-1}(c/2), f_{k-1}(c/4), \ldots f_{k-1}(c/2^\alpha)$), and the depth of the tree is $\log n^{1/\alpha}$. Say that the running time at the node $f_{\log n^{1/\alpha}}(n)$ is $T(n)$. Then because $f_k(n)$ must call each of $f_{k-1}(n/2^i)$ $n/2^i$ times for $1 \leq i \leq \alpha$, we have that $T(n) \leq \frac{\alpha n}{2}T(n/2)$. This recurrence yeilds $T(n) \leq poly(n)^{\log n} = O(2^{\log^2 n})$ for the running time.*

Generalizing our approach we can achieve a (slightly) better polynomial lowerbound on backlog. In our construction the point after which we had a bound for f_m grew further out by a factor of 2^9 each time. Instead of 2^9 we now use 2^α for some $\alpha \in \mathbb{N}$, and can find a better value of α. The value of α dictates how many itterations we can perform: we can perform $\log_2 n^{1/\alpha}$ itterations. The parameter α also dictates the multiplicative factor that we gain upon going from f_m to f_{m+1}. For $\alpha = 9$ this was 2. In general it turns out to be $\frac{\alpha-1}{4}$. Hence, we can achieve backlog $\Omega\left(\left(\frac{\alpha-1}{4}\right)^{\log_2 n^{1/\alpha}} \log_2 n\right)$. This optimizes at $\alpha = 13$, to backlog $\Omega(n^{\frac{\log_2 3}{13}} \log n) \approx \Omega(n^{0.122}\log n)$.

We can even improve over this. Note that in the proof that f_{m+1} gains a factor of 2 over f_m given above, we lowerbound $9\log_2 k - 9 \cdot 10/2$ with $2\log_2 k$. Usually however this is very loose: for small m a significant portion of the $9\log_2 k$ is annihlated by the constant $1 + 2 + \cdots + 9$ (or in general $\alpha \log_2 k$ and $1 + 2 + \cdots + \alpha$), but for larger values of m because k must be large we can get larger factors between steps, in theory factors arbitrarily close to α. If we could gain a factor of α at each step, then the backlog achievable would be $\Omega(\alpha^{\log_2 n^{1/\alpha}} \log n) = \Omega(n^{(\log_2 \alpha)/\alpha} \log n)$ which optimizes (over the naturals) at $\alpha = 3$ to $n^{(\log_2 3)/3} \approx n^{0.528}$. However, we can't actually gain a factor of α each time because of the subtracted constant. But, for any $\epsilon > 0$ we can achieve a $\alpha - \epsilon$ factor increase each time (for sufficiently large m). Of course ϵ can't be made arbitrarily small becasue m can't be made arbitrarily large, and the "cut off" m where we start achieving the $\alpha - \epsilon$ factor increase must be a constant (not dependent on n). When the cutoff m, or equivalently ϵ, is constant then we can achieve backlog $\Omega((\alpha-\epsilon)^{\log_2 n^{1/\alpha}} \log n) = \Omega(n^{(\log_2(\alpha-\epsilon))/\alpha} \log n)$. For instance, with this method we can get backlog $\Omega(\sqrt{n})$ for appropriate ϵ, α choice, or $\tilde{\Omega}(n^{(\log_2(3-\epsilon))/3})$ for any constant $\epsilon > 0$.

We could potentially aim to achieve even higher backlog by using more than the first α terms of the sum. The terms after $f_m(k/2^\alpha)$ in the sum are evaluated at points where they are potentially positive, but will not have the full strength of the $2^m \log_2 k$. This makes them difficult to deal with, and as it seems that we will just get a modest increase in the exponent of our polynomial we do not pursue this. □

Oblivious Lowerbounds.

An important theorem that we use repeatedly in our analysis is Hoeffding's Inequality:

Theorem 1 (Hoeffding's Inequality). *Let X_i be independent bounded random variabels with $X_i \in [a, b]$. Then,*

$$P\left(\left|\frac{1}{n}\sum_{i=1}^{n}(X_i - \mathbb{E}[X_i])\right| \geq t\right) \leq 2\exp\left(-\frac{2nt^2}{(b-a)^2}\right)$$

Proposition 2. *There exists an oblivious filling strategy in the variable-processor cup game on n cups that achieves backlog $\Omega(\log n)$ against a smoothed greedy emptier with probability at least $1 - 1/\operatorname{polylog}(n)$.*

Proof. Let A be the anchor set, randomly chosen, let B be non-anchor set, with $|A| = |B| = n/2$. Let $h = \Theta(1)$ be the fill that we will achieve at each level of our recursive procedure. Our strategy to achieve backlog $\Omega(\log n)$ is roughly as follows:

- Repeatedly make cups in B have a constant probability of having fill at least h, and then transfer these cups into A.
- Reduce the number of processors to a constant fraction nc of n and raise the fill of nc cups to $h/2$. This step relies on the emptier being greedy.

- Recurse on the nc cups that are known to have fill $\geq h/2$.

We can perfom $\Omega(\log n)$ levels of recursion, achieving constant backlog at each step (relative to average fill); doing so yields backlog $\Omega(\log n)$.

Our strategy is somewhat complicated by the possibility of the fill being very concentrated in a few cups. We proceed as follows:

For each anchor cup i we perform the following ***switching-process***:

1. Chose an index $j \in [n^2]$; our process proceeds for n^2 ***rounds***, j is the index of the switching-process at which we will switch a cup into the anchor set.

2. For n^2 rounds, we select a random subset $C \subset B$ of the non-anchor cups and play a single processor cup game on C.

3. On round j with $1/2$ probability we swap the winner of the single processor cup game into the anchor set, and with $1/2$ probability we swap a random cup from B into the anchor set.

Say that a cup is ***op*** if it contains fill $\geq \sqrt{\frac{nh}{\log\log n}}$. If there is ever an overpowered cup, then we win. Note that we don't need to know which cup is overpowered because it will take $\Omega(\text{poly}(n))$ rounds for the emptier to reduce the fill below $\text{poly}(n)$. Hence, we can assume without loss of generality that no cup is ever op.

We consider two cases:

- **Case 1:** For at least $1/2$ of the switching-processes, at least $1/2$ of the cups in B have fill $\geq -h/2$.

- **Case 2:** For at least $1/2$ of the switching-processes, less than $1/2$ of the cups in B have fill $\geq -h/2$.

Claim 1. *In Case 1, with probability at least* $1 - e^{-\Omega(n)}$*, we achieve fill* $\geq h/2$ *in a constant fraction of the cups in* A*.*

Proof. Consider a switching-process where we have at least $1/2$ of the cups in B have fill $\geq -h/2$.

Say the emptier ***neglects*** the anchor set in a round if on at least one step of the round the emptier does not empty from every anchor cup. By playing the single-processor cup game for n^2 rounds, with only one time when we actually swap a cup into the anchor set, we strongly disincentivise the emptier from neglecting the anchor set on more than a constant fraction of the rounds. The emptier must have some binary function, $I(k)$ that indicates whether or not they will neglect the anchor set on round k if we have not already swapped. Note that the emptier will know when we perform a swap, so whether or not the emptier neglects a round k depends on this information. This is the only relevant statistic that the emptier can use to decide whether or not to neglect a round, because on any round when we simply redistribute water amongst the non-anchor cups we effectively have not changed anything about the game state.

If the emptier is willing to neglect the anchor set for at least half of the rounds, i.e. $\sum_{k=1}^{n^2} I(k) \geq n^2/2$, then with probability at least $1/4$, $j \in (3/4n^2, n^2)$, so the emptier neglects the anchor set on at least $n^2/4$ rounds. The anchor set's average fill increases by at least 1 each round that the anchor set is neglected, and thus the anchor set's average fill will have increased by at least $n \geq \Omega(\text{poly}(n))$. Hence we have the desired backlog.

Otherwise, we have at least a $1/2$ chance that the round j, which is chosen uniformly at random from the rounds, when we will perform a switch into the anchor set occurs on a round with $I(j)$ indicating that the emptier won't neglect the anchor set on round j. In this case, the round was a legitimate single processor cup game on C_j, the randomly chosen set of e^h cups on the j-th round. Then we achieve fill increase $\geq h$ by the end of the game with probability at least $1/e^h!$, the probability that we correctly guess the sequence of cups within the single processor cup game that the emptier would empty from.

The probability that the random set $C_j \subset B$ contains only elements with fill $\geq -h/2$ is at least $1/2^{e^h}$, because at least half of the elements of B have fill $\geq -h/2$. If all elements of C_j have fill $\geq -h/2$, then the fill of the winner of the cup game has fill at least $-h/2 + h = h/2$ if we guess the emptier's emptying sequence correctly.

Combining the results, we have that for such a switching-process there is a constant probability of the cup which we switch into the anchor set has fill $\geq h/2$.

Say that this probability is $k \in (0, 1)$. Then the expectation of the number of cups in A with fill $\geq h/2$ is at least $kn/2$. Let X_i independent and identically distributed binary random variables, with X_i taking value 1 if a uniformly randomly selected element of A has fill $\geq h/2$ and X_i taking value 0 otherwise. Then by a Chernoff Bound (Hoeffding's Inequality applied to Binary Random Variables),

$$P\left(\sum_{i=1}^{n/2} X_i \le nk/4\right) \le e^{-n(k/2)^2}.$$

That is, the probability is exponentially small in n. $\square$

Claim 2. *In Case 2, with probability at least $1 - 1/\operatorname{polylog}(n)$, we achieve positive tilt $hn/8$ in the anchor set.*

Proof. Consider a switching-process where we have less than $1/2$ of the cups in B with fill $\ge -h/2$.

We assume for simplicity that the average fill of B is 0. In reality this is not the case, but by a Hoeffding bound and the fact that overpowered cups don't exist, the fill is really tightly concentrated around 0, so this is almost WLOG.

Let the positive tilt of a cup i be $\text{tilt}_+(i) := \max(\text{fill}(i), 0)$. We have

$$\mathbb{E}[\text{tilt}_+(X)] = \frac{1}{2}\,\mathbb{E}[|\,\text{fill}(X)|] \ge h/2$$

(because negative tilt is at least $nh/4$ and positive tilt must oppose this).

Let Y_i be the random variable $Y_i = \text{tilt}_+(X)$ where X is a randomly selected cup from the non-anchor set at the start of the i-th round of playing single processor cups games. Note that the Y_i are not really independent, but it is probably ok. Note that $0 \le Y_i \le hn/\lg\lg n$. Now we have, by Hoeffding's inequality, that

$$P\left(\Big|\frac{1}{n/2}\sum_{i=1}^{n/2}(Y_i - \mathbb{E}[Y_i])\Big| \ge h/4\right) \le$$

$$2\exp\left(-\frac{n(h/4)^2}{(\sqrt{hn/\lg\lg n})^2}\right)$$

$$P\left(\frac{1}{n/2}\sum_{i=1}^{n/2} Y_i \le h/4\right) \le 1/\operatorname{polylog}(n)$$

$\square$

Now we consider two cases based on how many times we must apply Claim 1 and Claim 2. If we must apply Claim 1 at least half the time, then we achieve a constant fraction of the anchor cups with fill at least $h/2$. If on the other hand we must apply Claim 2 at least half of the time, we have that with probability $1 - 1/\operatorname{polylog}(n)$ the process brings $n \cdot h/8$ positive tilt to the anchor set as desired.

In either case we achieve, with probability at least $1 - 1/\operatorname{polylog} n$, positive tilt at least hn/k in the anchor set. Use the positive tilt, with one processors, we can transfer over the fill into n/k cups. (Note, we use one processor because we do not know how many cups the fill is concentrated in). The filler repeatedly distributes 1 unit of fill to each of the n/k cups in succession, and continues until $h/4$ fill has been distributed. We cannot continue beyond this point because we have used up the positive tilt. Now we recurse on this set of n/k cups.

We can perform $\Omega(\log n)$ levels of recursion, and gain $\Omega(1)$ fill at each step. Hence, overall, backlog of $\Omega(\log n)$ is achieved.

Note that the only part of this proof that was specific to smoothed greedy was the end, when we wanted to achieve known fill in some cups. Against an arbitrary opponent we cannot assume that just because they are far behind means that they won't oppose our attempts to achieve cups with known fill. Extending this result to non-greedy emptiers, or showing that it cannot be extended is an important open question. $\square$

Lemma 2 (The Oblivious Amplification Lemma)**.** *Given an oblivious filling strategy for achieving backlog $f(k)$ in the variable-processor cup game on k cups that succeeds with probability at least $1/2$, there exists a strategy for achieving "amplified" fill*

$$f'(k) \ge \frac{1}{32}(f(k/2) + f(k/4) + f(k/8) + \cdots$$

that succceeds with constant probability.

Proof. We essentially perform the same proof as Proposition 2, but some new issues arise, which we proceed to highlight and address.

Claim 3. *Let a cup be **verysad** if it has fill $< -nh/\lg\lg n$. WLOG there are no verysad cups.*

Proof. First note that becasue WLOG there are no overpowered cups, there fewer than $n/2$ verysad cups.

Consider 2 cases:

- If the mass of the verysad cups is less than $nh/8$ then we can ignore them and accept a $-h/8$ penalty to the average fill.
- On the other hand, if the mass of the verysad cups is greater than $nh/8$, then by the end the average fill of everything else is already $h/8$ which is also basically as desired.

$\square$

Claim 4. *WLOG A, B have average fill $\geq -h/8$. In particular, we can construct a subset of $n/2$ cups with average fill $\geq -h/8$ with high probability in n.*

Proof. Recall the definition of an overpowered cup as a cup with fill $\geq nh/\lg\lg n$, and the fact that WLOG there are no overpowered cups. So, If we randomly pick B then this means that we are pretty good. Formalizing this, let X_i be the fill of the $n/2$-th randomly chosen cup for B. Unfortunately these are not quite independent events.

Lets say we pick $2n$ things from n things with replacement. Claim: with exponentially good probability we have $n/2$ distinct things. Proof: chernoff bound. Let X_i be indicator variable for cup i (whether it was chosen or not). Probability that X_i was chosen: $1-((n-1)/n)^n \approx 1-1/e > 1/2$ for large n. Then by a Chernoff Bound we have that $\sum_i X_i$ is tightly concentrated around its mean, which is larger than n. In particular, with probability exponentially close to 1 in n we have that at least $n/2$ cups were chosen.

initially solution: no overpowered cups wlog, so if we pick them randomly star holds by Hoeffding's. (kinda, bc stuff isnt really independent, can probably swap with replacement to fix this tho)

□

Claim 5. *What if C needs to be big because we need big backlog?*

Proof. this isnt a problem beause the base case is the only case that needs to explicitely deal with positive and negative fill □

These concerns resolved, the exact same argument as in Proposition 2 gives the desired result.

□

Corollary 2. *There is an oblivious filling strategy for the variable-processor cup game on n cups that achieves backlog $2^{\Omega(\sqrt{\log n})}$ in running time $O(n)$*

Proof. We must reduce want to reduce $\log^2 n$ to $\log n$ to achieve the appropriate running-time, so we reduce n to $n' = 2^{\sqrt{\log n}}$. This detail taken care of we apply exactly the same recursive construction of $f_{\theta(\log n)}$ as in Corollary 1, but using repeated application of the Oblivious Amplification Lemma rather than the Adaptive Amplification Lemma, which yields the disclaimer that the backlog is only achieved with constant probability. So we achieve backlog $\Omega(2^{\log n'})$ in running time $O(2^{\log^2 n'})$. By design, expressing this in terms of n we have running time $O(n)$ (randomized lowerbounds are not supposed to take longer than poly(n) time), and as a consequence we get backlog $\Omega(2^{\sqrt{\log n}})$. □