Balanced Sublists

1 Understanding the Balanced Function

The `balanced` function uses the tail-recursive auxiliary function `bal` to do its work. `bal` has three arguments:

- ms, which is the suffix of the overall input, ns, that remains to be processed;
- xs, which is the reversal of the longest nice (having no nonempty sublists summing to 0) suffix of the part of ns that's already been processed; and
- zss, which is all the balanced sublists of the part of ns that's already been processed, listed in strictly ascending order.

To begin with, $ms = ns$, $xs = []$ and $zss = []$, and when ms becomes empty, zss is `balanced`'s answer.

When ms is nonempty, the `extend` function is used to process its next element, m. The function `scanZeroSum` is first used to look for a prefix of xs such that the sum of m and the prefix's sum is 0.

- If there is no such prefix, then `extend` communicates back to `bal` that no new balanced sublists of ns have been found, and that the new version of xs will be $m :: xs$, and then `bal` iterates, replacing ms with its tail, xs with $m :: xs$, and leaving zss unchanged. Only one list cell needed to be created in this case, but length of xs additions were needed. All of the processing is done tail-recursively. In the worst case, xs can grow to be as long as ns.
- On the other hand, if there is a prefix ys of xs such that m plus the sum of ys is 0, then `extend` communicates back to `bal` that the reversal of $m :: ys$ is a newly found balanced sublist of ns and that all but the last element of $m :: ys$ should be the new version of xs, and then `bal` iterates, replacing ms with its tail, xs with all but the last element of $m :: ys$, and adding the reversal of $m :: ys$ to zss. It takes two times the length of ys plus 2 list cell allocations to build the new version of xs and find the new element of zss. And all of this processing is done tail-recursively, except for the insertion of the new answer into zss. If ys is short (when $m = 0$, it will be empty), then the new version of xs will also be short, making subsequent steps more efficient.

As a first example, let's consider how `balanced` works with input $ns = [1; 2; 3; 4; 5; -12; 3; 4; 5; -15]$. We start out with

$$ms = [1; 2; 3; 4; 5; -12; 3; 4; 5; -15], \qquad xs = [], \qquad zss = [].$$

The first five steps simply add elements to the front of xs. First, 1 is added, because there is no prefix of xs whose sum plus 1 is 0,

$$ms = [2; 3; 4; 5; -12; 3; 4; 5; -15], \qquad xs = [1], \qquad zss = [],$$

Next 2 is added,

$$ms = [3; 4; 5; -12; 3; 4; 5; -15], \qquad xs = [2; 1], \qquad zss = [].$$

Then 3 is added,

$$ms = [4; 5; -12; 3; 4; 5; -15], \qquad xs = [3; 2; 1], \qquad zss = [\,].$$

Then 4 is added,

$$ms = [5; -12; 3; 4; 5; -15], \qquad xs = [4; 3; 2; 1], \qquad zss = [\,].$$

And finally 5 is added,

$$ms = [-12; 3; 4; 5; -15], \qquad xs = [5; 4; 3; 2; 1], \qquad zss = [\,].$$

Because the next element of ms is now -12, and $[5; 4; 3]$ is a prefix of xs whose sum plus -12 is 0, in the next step, xs becomes all but the last element of $[-12; 5; 4; 3]$, and the reversal of $[-12; 5; 4; 3]$ is inserted into zss,

$$ms = [3; 4; 5; -15], \qquad xs = [-12; 5; 4], \qquad zss = [[3; 4; 5; -12]].$$

Because the next element of ms is now 3, and $[-12; 5; 4]$ is a prefix of xs whose sum plus 3 is 0, in the next step, xs becomes all but the last element of $[3; -12; 5; 4]$, and the reversal of $[3; -12; 5; 4]$ is inserted into zss,

$$ms = [4; 5; -15], \qquad xs = [3; -12; 5], \qquad zss = [[3; 4; 5; -12]; [4; 5; -12; 3]].$$

Because the next element of ms is now 4, and $[3; -12; 5]$ is a prefix of xs whose sum plus 4 is 0, in the next step, xs becomes all but the last element of $[4; 3; -12; 5]$, and the reversal of $[4; 3; -12; 5]$ is inserted into zss,

$$ms = [5; -15], \qquad xs = [4; 3; -12], \qquad zss = [[3; 4; 5; -12]; [4; 5; -12; 3]; [5; -12; 3; 4]].$$

Because the next element of ms is now 5, and $[4; 3; -12]$ is a prefix of xs whose sum plus 5 is 0, in the next step, xs becomes all but the last element of $[5; 4; 3; -12]$, and the reversal of $[5; 4; 3; -12]$ is inserted into zss,

$$ms = [-15], \qquad xs = [5; 4; 3], \qquad zss = [[-12; 3; 4; 5]; [3; 4; 5; -12]; [4; 5; -12; 3]; [5; -12; 3; 4]].$$

Because the next element of ms is now -15, and there is no prefix of xs whose sum plus -15 is 0, in the next step, -15 is simply added to the front of xs,

$$ms = [\,], \qquad xs = [-15; 5; 4; 3], \qquad zss = [[-12; 3; 4; 5]; [3; 4; 5; -12]; [4; 5; -12; 3]; [5; -12; 3; 4]].$$

Finally, ms is empty, so the result is zss.

As a second example, let's consider how `balanced` works with input $ns = [1; 2; 3; 0; -3; -2; -1]$. We start out with

$$ms = [1; 2; 3; 0; -3; -2; -1], \qquad xs = [], \qquad zss = [].$$

First, we add 1 to xs,

$$ms = [2; 3; 0; -3; -2; -1], \qquad xs = [1], \qquad zss = [].$$

Next, we add 2 to xs,

$$ms = [3; 0; -3; -2; -1], \qquad xs = [2; 1], \qquad zss = [].$$

Next, we add 3 to xs,

$$ms = [0; -3; -2; -1], \qquad xs = [3; 2; 1], \qquad zss = [].$$

Because the next element of ms is now 0, and $[\,]$ is a prefix of xs whose sum plus 0 is 0, in the next step, xs becomes all but the last element of $[0]$, and the reversal of $[0]$ is inserted into zss,

$$ms = [-3; -2; -1], \qquad xs = [], \qquad zss = [[0]].$$

The rest of the steps add -3, -2 and -1 to xs, without adding anything to zss.

2 Proving Correctness of the Balanced Function

In what follows, we'll abbreviate `List.rev`, `List.hd` and `List.length` to `rev`, `hd` and `length`, respectively. `length` is defined so that, for all lists xs and ys of values of the same type, $\texttt{length}(xs \mathbin{@} ys) = \texttt{length}\, xs + \texttt{length}\, ys$. And `rev` is defined so that, for all lists xs and ys of values of the same type, $\texttt{rev}(xs \mathbin{@} ys) = \texttt{rev}\, ys \mathbin{@} \texttt{rev}\, xs$. We'll use the following lemma about `rev` repeatedly and without citation:

Lemma 2.1

(1) For all lists xs, $\texttt{rev}(\texttt{rev}\, xs) = xs$.

(2) For all lists xs and ys of values of the same type, $xs = ys$ iff $\texttt{rev}\, xs = \texttt{rev}\, ys$.

(3) For all lists xs and ys of values of the same type, $xs = \texttt{rev}\, ys$ iff $\texttt{rev}\, xs = ys$.

(4) For all lists xs, $\texttt{length}\, xs = \texttt{length}(\texttt{rev}\, xs)$.

Proof. (1) can be proved by induction on xs. (2) and (3) follow immediately from (1). And (4) follows by induction on xs. $\square$

Lemma 2.2

For all nice lists xs and suffixes us and vs of xs, if us and vs have the same sum, then $us = vs$.

Proof. Suppose xs is a nice list and us and vs are suffixes of xs with the same sum. We must show that $us = vs$. We'll consider the case where us is no longer than vs, the other case being symmetric. Thus $vs = ws \mathbin{@} us$ for some ws. Since ws is a sublist of vs, which is a sublist of xs, we have that ws is a sublist of xs. Since us and vs have the same sum, and the sum of vs is the sum of ws plus the sum of us, it follows that ws has a sum of 0. Thus, because xs is nice and ws is a sublist of xs, we have that $ws = [\,]$, so that $us = [\,] \mathbin{@} us = ws \mathbin{@} us = vs$. $\square$

Lemma 2.3

There is at most one balanced suffix of a list of integers ns.

Proof. Suppose us and vs are balanced suffixes of a list of integers ns. We must show that they are equal. We'll consider the case where us is no longer than vs, the other case being symmetric. Thus $vs = ws \mathbin{@} us$ for some ws. Because us and vs both sum to 0, the sum of ws must also be 0. Thus, since $ws \mathbin{@} us = vs$ is balanced, we have that either $ws = [\,]$ or $us = [\,]$, as otherwise both of ws and us would be proper, nonempty sublists of vs with sum 0. In the first case, we have that $us = [\,] \mathbin{@} us = ws \mathbin{@} us = vs$. And, in the second case, we have that $[\,] = us$ is balanced—contradiction. Thus $us = vs$. □

Lemma 2.4
If xs is a nice list, n is a nonzero integer, and the sum of xs plus n is 0, then $xs \mathbin{@} [n]$ is balanced.

Proof. Suppose xs is a nice list, n is a nonzero integer, and the sum of xs plus n is 0. We must show that $xs \mathbin{@} [n]$ is balanced. Because it clearly has a sum of 0 and is nonempty, it remains to show that it has no proper, nonempty sublist with sum 0. Suppose, toward a contradiction, that us is a proper, nonempty sublist of $xs \mathbin{@} [n]$ with sum 0. There are two cases to consider.

- Suppose us is a sublist of xs. Because us is nonempty and sums to 0, and xs is nice, we have a contradiction.
- Suppose us is a suffix of $xs \mathbin{@} [n]$. Because us is nonempty, there is a suffix vs of xs such that $us = vs \mathbin{@} [n]$. Because us sums to 0, it follows that vs sums to $-n$. But the sum of xs plus n is 0, and thus xs also sums to $-n$. Because xs and vs are suffixes of the same nice list (xs) and have the same sum, Lemma 2.2 tells us that $xs = vs$. But then $us = vs \mathbin{@} [n] = xs \mathbin{@} [n]$, contradicting that us is a proper sublist of $xs \mathbin{@} [n]$.

□

Lemma 2.5
Suppose ns and xs are lists of integers. The following statements are equivalent:

(1) the reversals of the prefixes of xs are the nice suffixes of ns;

(2) $\texttt{rev}\,xs$ is the longest nice suffix of ns.

Proof. Suppose that ns and xs are lists of integers.

- ((1) implies (2)) Suppose the reversals of the prefixes of xs are the nice suffixes of ns. We must show that $\texttt{rev}\,xs$ is the longest nice suffix of ns. Because xs is a prefix of itself, $\texttt{rev}\,xs$ is a nice suffix of ns. Suppose, toward a contradiction, that there is nice suffix ms of ns that is longer than $\texttt{rev}\,xs$. Then $ms = \texttt{rev}\,ys$ for some prefix ys of xs. Because $\texttt{rev}\,ys = ms$ is longer than $\texttt{rev}\,xs$, we have that $ys = \texttt{rev}(\texttt{rev}\,ys)$ is longer than $xs = \texttt{rev}(\texttt{rev}\,xs)$—contradicting the fact that ys is a prefix of xs. Thus $\texttt{rev}\,xs$ is the longest nice suffix of ns.
- ((2) implies (1)) Suppose $\texttt{rev}\,xs$ is the longest nice suffix of ns. We must show that the reversals of the prefixes of xs are the nice suffixes of ns.

  First, suppose that ys is a prefix of xs. Then $xs = ys \mathbin{@} us$ for some us, so that $\texttt{rev}\,us \mathbin{@} \texttt{rev}\,ys = \texttt{rev}(ys \mathbin{@} us) = \texttt{rev}\,xs$. Because $\texttt{rev}\,ys$ is a suffix of $\texttt{rev}\,xs$, which is in turn a suffix of ns, we have that $\texttt{rev}\,ys$ is a suffix of ns. And since $\texttt{rev}\,ys$ is a sublist of the nice list $\texttt{rev}\,xs$, it follows that $\texttt{rev}\,ys$ is nice, as any nonempty sublist of $\texttt{rev}\,ys$ with a sum of 0 would also be a nonempty sublist of $\texttt{rev}\,xs$. Thus $\texttt{rev}\,ys$ is a nice suffix of ns.

Second, suppose that ms is a nice suffix of ns. Because $\texttt{rev}\,xs$ is the longest nice suffix of ns, it follows that ms is a suffix of $\texttt{rev}\,xs$. Thus there is a us such that $\texttt{rev}\,xs = us \mathbin{@} ms$, so that $xs = \texttt{rev}(\texttt{rev}\,xs) = \texttt{rev}(us \mathbin{@} ms) = \texttt{rev}\,ms \mathbin{@} \texttt{rev}\,us$. Hence $\texttt{rev}\,ms$ is a prefix of xs whose reversal is ms.

□

Correctness Proof for `extend`

Suppose ns is a list of integers, n is an integer, and the reversals of the prefixes of xs are the nice suffixes of ns. By Lemma 2.5, we have that $\texttt{rev}\,xs$ is the longest nice suffix of ns. There are two cases to consider.

- Suppose there is no prefix of xs whose sum, when added to n, yields 0. Then $\texttt{scanZeroSum}\,n\,xs$ returns $\texttt{None}$, so that $\texttt{extend}\,n\,xs$ returns $(\texttt{None}, n :: xs)$. Thus we must show that:

  (1) there is no balanced suffix of $ns \mathbin{@} [n]$, and

  (2) the reversals of the prefixes of $n :: xs$ are the nice suffixes of $ns \mathbin{@} [n]$, which, by Lemma 2.5, is equivalent to showing that $\texttt{rev}\,xs \mathbin{@} [n] = \texttt{rev}(n :: xs)$ is the longest nice suffix of $ns \mathbin{@} [n]$.

  We have that $n \neq 0$, since otherwise $[\,]$ would be a prefix of xs whose sum, when added to n, yields 0.

  Suppose, toward a contradiction, that there is a balanced suffix of $ns \mathbin{@} [n]$. Because balanced lists are nonempty, it follows that there is a suffix ms of ns such that $ms \mathbin{@} [n]$ is balanced. Because ms is a proper sublist of $ms \mathbin{@} [n]$, it follows that ms is nice (a nonempty sublist of ms with sum 0 would be a proper, nonempty sublist of $ms \mathbin{@} [n]$). Thus, since ms is a nice suffix of ns, we have that $\texttt{rev}\,ms$ is a prefix of xs. Because $ms \mathbin{@} [n]$ is balanced, we have that the sum of ms, when added to n, yields 0. But then the sum of $\texttt{rev}\,ms$, when added to n, also yields 0, so that there is prefix of xs whose sum, when added to n, yields 0—contradiction. Thus there is no balanced suffix of $ns \mathbin{@} [n]$, i.e., (1) holds.

  It remains to show (2). Suppose, toward a contradiction, that $\texttt{rev}\,xs \mathbin{@} [n]$ is not nice. Because $n \neq 0$ and $\texttt{rev}\,xs$ is nice, it follows that there is a nonempty suffix us of $\texttt{rev}\,xs$ such that the sum of us, when added to n, yields 0. Because us is a sublist of $\texttt{rev}\,xs$, and $\texttt{rev}\,xs$ is nice, we have that us is nice. Thus Lemma 2.4 tells us that $us \mathbin{@} [n]$ is balanced. Since us is a suffix of $\texttt{rev}\,xs$, and $\texttt{rev}\,xs$ is a suffix of ns, we have that us is a suffix of ns, so that $us \mathbin{@} [n]$ is a balanced suffix of $ns \mathbin{@} [n]$—contradiction. Thus we have that $\texttt{rev}\,xs \mathbin{@} [n]$ is a nice suffix of $ns \mathbin{@} [n]$.

  Finally, suppose, toward a contradiction, that there is a nice suffix us of $ns \mathbin{@} [n]$ that is longer than $\texttt{rev}\,xs \mathbin{@} [n]$. Then $us = ms \mathbin{@} [n]$, where ms is a longer suffix of ns than $\texttt{rev}\,xs$. Because us is nice and ms is a sublist of us, we have that ms is nice, so that ms is a nice suffix of ns that is longer than $\texttt{rev}\,xs$—contradiction. This concludes the proof of (2).

- Suppose there is a prefix of xs whose sum, when added to n, yields 0. Let ys be the shortest such prefix. Then $\texttt{scanZeroSum}\, n\, xs$ returns $\texttt{Some}\, i$, where i is the length of ys. Since $0 \leq i$ and i is less-than-or-equal-to the length of $n :: xs$, $\texttt{splitRev}\, i\, (n :: xs)$ returns (us, vs), where us is the reversal of the first i elements of $n :: xs$, and vs is the remaining elements of $n :: xs$. Thus $\texttt{extend}\, n\, xs$ returns $(\texttt{Some}(\texttt{hd}\, vs :: us), \texttt{rev}\, us)$.

  Because i is the length of ys, and ys is a prefix of xs, it follows that $\texttt{rev}\, us$ is all but the last element of $n :: ys$, and that $\texttt{hd}\, vs$ is the last element of $n :: ys$. Thus $n :: ys = \texttt{rev}\, us \mathbin{@} [\texttt{hd}\, vs]$. Hence $\texttt{rev}(n :: ys) = \texttt{rev}(\texttt{rev}\, us \mathbin{@} [\texttt{hd}\, vs]) = \texttt{rev}[\texttt{hd}\, vs] \mathbin{@} \texttt{rev}(\texttt{rev}\, us) = [\texttt{hd}\, vs] \mathbin{@} us = \texttt{hd}\, vs :: us$. Thus $\texttt{extend}\, n\, xs$ returns $\texttt{Some}(\texttt{rev}(n :: ys))$ paired with all but the last element of $n :: ys$.

  Hence we must show that:

  (1) $\texttt{rev}(n :: ys)$ is the unique balanced suffix of $ns \mathbin{@} [n]$, which, by Lemma 2.3, is equivalent to showing that $\texttt{rev}\, ys \mathbin{@} [n] = \texttt{rev}(n :: ys)$ is a balanced suffix of $ns \mathbin{@} [n]$, and

  (2) the reversals of the prefixes of all but the last element of $n :: ys$ are the nice suffixes of $ns \mathbin{@} [n]$, which, by Lemma 2.5, is equivalent to showing that the reversal of all but the last element of $n :: ys$ is the longest nice suffix of $ns \mathbin{@} [n]$.

  We know that n plus the sum of ys is 0, and thus that $\texttt{rev}\, ys \mathbin{@} [n]$ has a sum of 0. And, clearly $\texttt{rev}\, ys \mathbin{@} [n]$ is nonempty. So for (1) it remains to show that $\texttt{rev}\, ys \mathbin{@} [n]$ has no proper, nonempty sublist with a sum of 0. To this end, suppose, toward a contradiction, that zs is a proper, nonempty sublist of $\texttt{rev}\, ys \mathbin{@} [n]$ with a sum of 0. Because ys is a prefix of xs, $\texttt{rev}\, ys$ is a nice suffix of ns. Thus zs is not a sublist of $\texttt{rev}\, ys$, so that $zs = ws \mathbin{@} [n]$ for some suffix ws of $\texttt{rev}\, ys$. Since the sum of zs is 0, we have that the sum of ws is $-n$. But, since the sum of $\texttt{rev}\, ys \mathbin{@} [n]$ is 0, we also have that the sum of $\texttt{rev}\, ys$ is $-n$. Because $\texttt{rev}\, ys$ is nice, $\texttt{rev}\, ys$ and ws are suffixes of $\texttt{rev}\, ys$, and $\texttt{rev}\, ys$ and ws have identical sums, Lemma 2.2 tells us that $\texttt{rev}\, ys = ws$. But this means that $zs = ws \mathbin{@} [n] = \texttt{rev}\, ys \mathbin{@} [n]$, showing that zs is not a proper sublist of $\texttt{rev}\, ys \mathbin{@} [n]$—contradiction. This completes the proof of (1).

  For (2), there are two cases to consider.

  - Suppose $n = 0$. Then $ys = [\,]$, by its definition. Hence all but the last element of $n :: ys$ is $[\,]$. So we must show that $[\,] = \texttt{rev}[\,]$ is the longest nice suffix of $ns \mathbin{@} [n]$. $[\,]$ is nice and is a suffix of any list. Furthermore, any longer suffix of $ns \mathbin{@} [n]$ would have $[0]$ as a sublist, and so wouldn't be nice. So $[\,]$ is the longest nice suffix of $ns \mathbin{@} [n]$.
  - Suppose $n \neq 0$. Then the sum of ys is $-n$, so that $ys = us \mathbin{@} [m]$ for some us and m. We must show that $\texttt{rev}\, us \mathbin{@} [n] = \texttt{rev}(n :: us)$ is the longest nice suffix of $ns \mathbin{@} [n]$. We have that $\texttt{rev}\, ys \mathbin{@} [n] = \texttt{rev}(us \mathbin{@} [m]) \mathbin{@} [n] = m :: \texttt{rev}\, us \mathbin{@} [n]$. From (1), we know that $\texttt{rev}\, ys \mathbin{@} [n]$ is a balanced suffix of $ns \mathbin{@} [n]$. Since $\texttt{rev}\, us \mathbin{@} [n]$ is a suffix of $\texttt{rev}\, ys \mathbin{@} [n]$, we have that $\texttt{rev}\, us \mathbin{@} [n]$ is a suffix of $ns \mathbin{@} [n]$. Furthermore, because $\texttt{rev}\, us \mathbin{@} [n]$ is a proper sublist of the balanced list $\texttt{rev}\, ys \mathbin{@} [n]$, it follows that $\texttt{rev}\, us \mathbin{@} [n]$ is nice (if $\texttt{rev}\, us \mathbin{@} [n]$ had a nonempty sublist with sum 0, then $\texttt{rev}\, ys \mathbin{@} [n]$ would have a proper, nonempty sublist with sum 0). Furthermore, any longer suffix of $ns \mathbin{@} [n]$ would contain $m :: \texttt{rev}\, us \mathbin{@} [n] = \texttt{rev}\, ys \mathbin{@} [n]$ (which has a sum of 0) as a sublist, and so wouldn't be nice. Thus $\texttt{rev}\, us \mathbin{@} [n]$ is the longest nice suffix of $ns \mathbin{@} [n]$.

Correctness Proof for `balanced`

Suppose ns is a list of integers. First, we'll show that, under the assumption that the auxiliary function `bal` is correct, `balanced` ns returns the balanced sublists of ns, listed in strictly ascending order. Then, we'll prove that `bal` is correct.

We have that ns is a suffix of itself, and the prefix of ns of length $\texttt{length}\,ns - \texttt{length}\,ns$ is $[\,]$. Since the reversals of the prefixes of $[\,]$ (there is only one, $[\,]$) are the nice suffixes of $[\,]$ (there is only one, $[\,]$), we have that the reversals of the prefixes of $[\,]$ are the nice suffixes of the prefix of ns of length $\texttt{length}\,ns - \texttt{length}\,ns$. Since there are no balanced sublists of $[\,]$, we have that $[\,]$ is the balanced sublists of $[\,]$, listed in strictly ascending order, i.e., is the balanced sublists of the prefix of ns of length $\texttt{length}\,ns - \texttt{length}\,ns$, listed in strictly ascending order. Thus, by `bal`'s specification, $\texttt{bal}\,ns\,[\,]\,[\,]$ returns the balanced sublists of ns, listed in strictly ascending order. And this, in turn, is what $\texttt{balanced}\,ns$ returns.

To prove that `bal` is correct, suppose ms is a suffix of ns, and the reversals of the prefixes of xs are the nice suffixes of the prefix of ns of length $\texttt{length}\,ns - \texttt{length}\,ms$, and zss is the balanced sublists of the prefix of ns of length $\texttt{length}\,ns - \texttt{length}\,ms$, listed in strictly ascending order. We must prove that $\texttt{bal}\,ms\,xs\,zss$ returns all the balanced sublists of ns, listed in strictly ascending order. There are two cases to consider.

- Suppose $ms = [\,]$. Then the prefix of ns of length $\texttt{length}\,ns - \texttt{length}\,ms$ is ns, and so zss is the balanced sublists of ns, listed in strictly ascending order. But this is what `bal` returns.

- Suppose $ms = m :: ms'$ for some integer m and list of integers ms'. Because ms is a suffix of ns, we have that ms' is a suffix of ns. Let ls be the prefix of ns of length $\texttt{length}\,ns - \texttt{length}\,ms$. Then $ls \mathbin{@} [m]$ is the prefix of ns of length $\texttt{length}\,ns - \texttt{length}\,ms'$. There are two subcases to consider.

  - Suppose there is no balanced suffix of $ls \mathbin{@} [m]$. Because the reversals of the prefixes of xs are the nice suffixes of ls, we have that $\texttt{extend}\,m\,xs$ returns $(\texttt{None}, ys)$, where the reversals of the prefixes of ys are the nice suffixes of $ls \mathbin{@} [m]$. Because zss is the balanced sublists of ls, listed in strictly ascending order, and there is no balanced suffix of $ls \mathbin{@} [m]$, we have that zss is the balanced sublists of $ls \mathbin{@} [m]$, listed in strictly ascending order. Thus, by the specification of `bal`, we have that $\texttt{bal}\,ms'\,ys\,zss$ returns the balanced sublists of ns, listed in strictly ascending order. But this is what `bal` returns.

  - Suppose there is a balanced suffix of $ls \mathbin{@} [m]$. Because the reversals of the prefixes of xs are the nice suffixes of ls, we have that $\texttt{extend}\,m\,xs$ returns $(\texttt{Some}\,zs, ys)$, where zs is the unique balanced suffix of $ls \mathbin{@} [m]$, and the reversals of the prefixes of ys are the nice suffixes of $ls \mathbin{@} [m]$. Since the elements of zss are sorted in strictly ascending order, $\texttt{insert}\,zs\,zss$ consists of zs plus all of the elements in zss, listed in strictly ascending order. Because zss is the balanced sublists of ls, and zs is the unique balanced suffix of $ls \mathbin{@} [m]$, it follows that $\texttt{insert}\,zs\,zss$ is the balanced sublists of $ls \mathbin{@} [m]$, listed in strictly ascending order. Thus, by the specification of `bal`, we have that $\texttt{bal}\,ms'\,ys\,(\texttt{insert}\,zs\,zss)$ returns the balanced sublists of ns, listed in strictly ascending order. But this is what `bal` returns.