Oliver Moseler

Performance test 2

Comparison of the runtime between the two methods *retainAll(Collection coll)* and *addAll(int index, Collection coll)* of the class *org.apache.commons.collections.list.SetUniqueList*. Both need to iterate through its elements and need to check whether an item is already present or not. Hence their runtime should be similar.

The method *addAll(...)* is optimized already.

Task

Find the performance bug in method *retainAll* and try to understand it.

Problem is

Method retainAll(...) actually calls two times AbstractCollection.retainAll(...). This calls ArrayList.contains(...) which calls ArrayList.indexOf(...). This does a linear search on the list. Thus retainAll(...) has an asymptotic runtime of $\mathcal{O}(n^2)$.
Caution, ArrayList.contains(...) has a very high selftime. I guess method indexOf will get inlined at this point.

Solution

Try to not delegate the functionality to AbstractCollection and implement own functionality. Encapsulate collections into a HashSet so that the method contains(...) runs in $\mathcal{O}(1)$.

Hints

1. The functionality is delegated into AbstractCollection, two times.
2. The highest runtime impact on AbstractCollection.retainAll(...) has the callee method contains.
3. A lookup in $\mathcal{O}(1)$ can be done with the help of a HashSet.