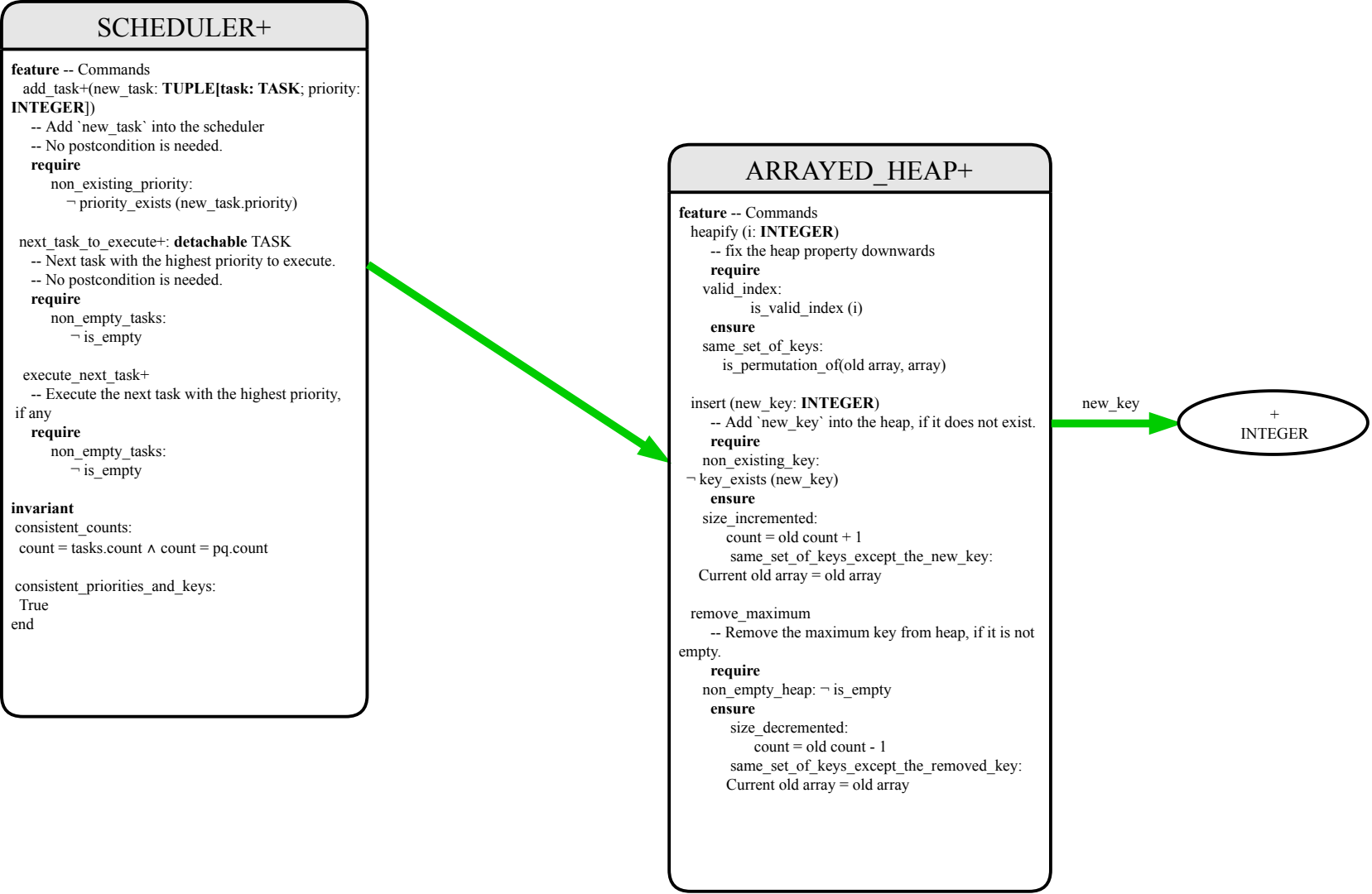
SCHEDULER+
feature -- Commands
add_task+(new_task: TUPLE[task: TASK; priority: INTEGER])
-- Add `new_task` into the scheduler
-- No postcondition is needed.
require
non_existing_priority:
¬ priority_exists (new_task.priority)
next_task_to_execute+: detachable TASK
-- Next task with the highest priority to execute.
-- No postcondition is needed.
require
non_empty_tasks:
¬ is_empty
execute_next_task+
-- Execute the next task with the highest priority, if any
require
non_empty_tasks:
¬ is_empty
invariant
consistent_counts:
count = tasks.count ∧ count = pq.count
consistent_priorities_and_keys:
True
end
ARRAYED_HEAP+
feature -- Commands
heapify (i: INTEGER)
-- fix the heap property downwards
require
valid_index:
is_valid_index (i)
ensure
same_set_of_keys:
is_permutation_of(old array, array)
insert (new_key: INTEGER)
-- Add `new_key` into the heap, if it does not exist.
require
non_existing_key:
¬ key_exists (new_key)
ensure
size_incremented:
count = old count + 1
same_set_of_keys_except_the_new_key:
Current old array = old array
remove_maximum
-- Remove the maximum key from heap, if it is not empty.
require
non_empty_heap: ¬ is_empty
ensure
size_decremented:
count = old count - 1
same_set_of_keys_except_the_removed_key:
Current old array = old array
new_key
+
INTEGER