Linked lists

Christophe Rhodes

Motivation

- simple application of pairs
- building block for more complex data structures:
  - stacks
  - queues
- useful for considering issues in algorithm design:
  - complexity and scaling
  - iteration and recursion

Definition

A linked list is a sequential collection of data

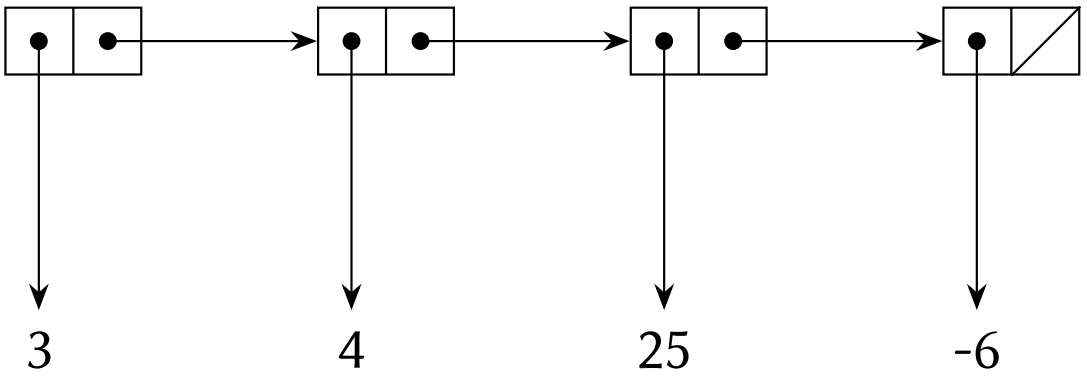
Operations

first	return the first element of the list
rest	return the list with the first element removed
cons[o]	return a new list whose first is o and whose rest is the list
set-first![o]	set the first of the list to o
set-rest![l]	set the rest of the list to l
null?	return `true` if the list is the *empty list*

Special value

NIL the immutable empty list

Implementation



Complexity analysis

first, rest, set-first!, set-rest!

1. pointer read (first, rest) or write (set-first!, set-rest!)

$$\Longrightarrow \Theta(1)$$

cons

1. fixed-size (two word) allocation
2. two pointer writes

$$\Longrightarrow \Theta(1)$$

null?

1. single comparison

$$\Longrightarrow \Theta(1)$$

Complexity analysis

construct a linked list

... with N elements

1. construct N nodes
2. set-first! N times
3. set-rest! $N - 1$ times

$$\Longrightarrow \Theta(N)$$

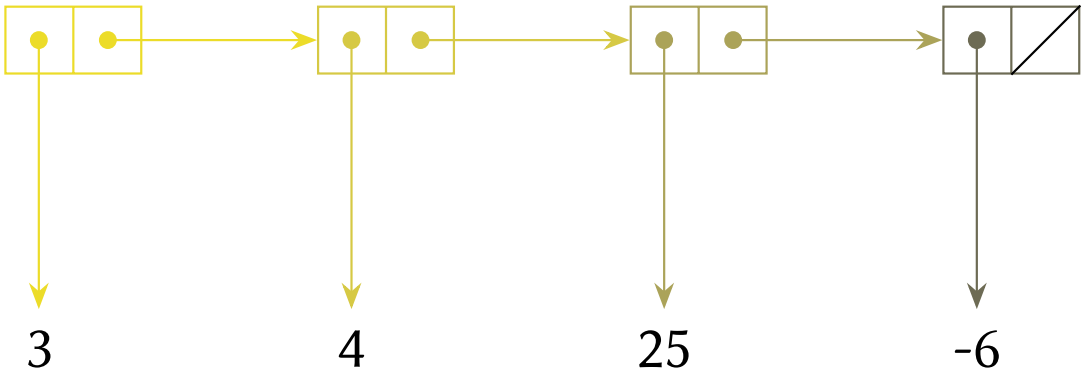
or

Let the time for constructing a linked list with k elements be T_k

1. construct a list of length $N - 1$: T_{N-1}
2. construct a node: $\Theta(1)$

$$\begin{aligned} \Longrightarrow T_N &= T_{N-1} + \Theta(1) \\ &= \Theta(N) \end{aligned}$$

Recursive Data Structure



Linear linked list algorithms

Base case

what is the answer for the empty list?

Otherwise

1. compute the answer for the rest of the list
2. modify that answer based on the current node

Example: length

Base case

what is the length of the empty list?

Otherwise

1. what is the length of the rest of the list?
2. how does the length of this list relate to the length of the rest of the list?