1. Introduction
   1. Linked list: “A linked list is a sequence of elements arranged one after another, with each, element connected to the next element by a “link.” A common programming, practice is to place each element together with the link to the next element,, resulting in a component called a node. A node is represented pictorially as a, box, with the element written inside the box and the link drawn as an arrow, pointing out of the box.” “These arrows, or links, are used to connect one node to another. “ (Main, p. 176)
   2. “The links are represented as arrows because they do more than simply connect, two nodes. The links also place the nodes in a particular order. In Figure 4.1, the, five nodes form a chain from top to bottom. The first node is linked to the second, node; the second node is linked to the third node; and so on, until we reach the, last node. We must do something special when we reach the last node since, the last node is not linked to another node. In this special case, we will replace, the link in this node with a note saying “end marker.”” (Main, p. 176)
2. Use in bioinformatics
   1. The linked-list is an ideal data structure for DNA sequences.
   2. “Assembling the, coding blocks from micronuclear genes to form functional macronuclear, genes is facilitated by an impressive in-vivo implementation of the linked, list data structure of computer science.” (Harju, T., Li, C., Petre, I. & Rozenberg, G. 2007)
   3. “The macronuclear gene is a contiguous DNA sequence, which is placed on, its own chromosome, that (with few exceptions only) is not shared with other, genes. The same gene in the micronucleus is broken into pieces called MDSs, (macronuclear destined sequences) that are separated by noncoding blocks called, IESs (internally eliminated sequences). Moreover, the order of MDSs may be, permuted (with respect to their order in the macronuclear gene), and some of the, MDSs may be inverted. Here is where the challenge of gene assembly lies: ciliates, have to identify correctly more than 100 000 MDSs in their genome, see [20],, assemble them together in the macronuclear (orthodox) order, and eliminate all, IESs. We refer to [12], [19], [23] for more details on ciliates and gene assembly., A hint on how ciliates achieve gene assembly is given by the structure of, MDSs. It turns out that ciliates organize their genomic data as linked lists in, the style used in computer science, see [19]. A short sequence at the end of, each MDS is repeated at the beginning of the MDS that should follow it in, the orthodox order, thus (in the terminology of computer science) serving as a, pointer in a linked list. It is currently believed that ciliates splice together the, consecutive MDSs on the common pointers to assemble the gene. The models, for gene assembly in Stichotrichs, such as, e.g., [16], [17] and [8], [21], agree on, this generic mechanism.” (Harju, T., Li, C., Petre, I. & Rozenberg, G. 2007)
   4. Assembling of DNA fragments. (DNA Fragment Assembly Using The Greedy Algorithm) (Li, L. & Khuri, S. 2004)
      1. “The major steps of our Greedy Algorithm are:”
         1. “Construction of BSC: The algorithm takes as input a set of fragments and outputs a best set of contigs stored in a vector template. The linked list template and the vector template are the major data structures used in this step. The linked list is composed of the overlap scores sorted in descending order. Each item contains a pair of fragment IDs and their overlap score. The first fragment is from the forward direction and the second fragment is the reverse complement of another fragment that has overlap length above some threshold with the first fragment. The vector represents a set of contigs. Each Contig object contains a pair of fragments and their overlap weight.” (Li, L. & Khuri, S. 2004)
   5. “DNA Fragment Assembly using the Clustering Heuristic Algorithm” (Li, L. & Khuri, S. 2004)
      1. main parts:
         1. “Construct a score table for all possible pairs of fragments considering forward directions and reverse complements.”
         2. “Sort the score in descending order and insert all FragmentPair objects (a pair whose score is above some threshold) into a linked list. Each FragmentPair node contains a pair of fragment IDs and overlap score. If the score is positive, it indicates that both fragments are from the same strand. A negative score means that the two fragments are from different strands.”
         3. “Select the first node (a,b) in the linked list as a starting point to order the rest of the fragments. Set a to be the first fragment and b to be the last fragment in the current layout.”
         4. “Select the next node in the linked list and compare the pair of fragments with the first and the last fragments in the current layout. If the clustering is successful, the fragment ID joins the set. If it needs to be inserted in the front, we reset the first fragment in the layout. If it needs to be appended at the end, we reset the last fragment in the layout. Otherwise, we put the node in a temporary sorted linked list. The process continues until the current linked list is traversed. Note that only one possible direction for each fragment can be chosen.”
         5. “When the current linked list is being traversed, a new contig is created. We need to remove the nodes that contain IDs in the selected fragment set from the temporary linked list. Next, we reset the temporary linked list as the current linked list and continue from Step 3 until no more fragments are left. Each contig contains the list of ordered fragment IDs.”

References:

Main, Michael (2011-12-01). Data Structures and Other Objects Using Java (2-download) (4th Edition) (Page 176). Prentice Hall. Kindle Edition.

Harju, T., Li, C., Petre, I. & Rozenberg, G. 2007, "Complexity measures for gene assembly" in Knowledge Discovery and Emergent Complexity in Bioinformatics Springer, , pp. 42-60.

Li, L. & Khuri, S. 2004, "A Comparison of DNA Fragment Assembly Algorithms.", METMBS, pp. 329.