Global Journal of Enterprise Information System

Utilisation of Segment for Displaying of DNA, RNA and Protein Related Images

Chintankumar Dayalal Gohel*

Assistant Professor, Department of Computer Science, Gujarat Vidyapith, Ashram Road, Ahmedabad, India; chintan.kumar@gmail.com

Abstract

Sometimes bioinformation may be composed of several pictures or items of information. A single image of bioinformation may contain several views of an object. It may have a picture of the overall object and a close up of a particular component. As an example suppose we have image for DNA, RNA and Protein. So, some user only wants to display DNA structure as well some other user wants to display only RNA, someone wants Protein and DNA structure. There is also case when transformations will be applied to only RNA while another structure of DNA will remain as it is. So, we wish to apply them to only a portion of the picture either for DNA or for RNA or Protein or combination from group of these three structures. So, it is required to organize display file to reflect this subpicture structure. For that the display file will be divided into segments where each segments corresponding to components of the overall display. A visible segment will be displayed but an invisible segment will not be shown.

Keywords: DNA, Protein, RNA, Segment, Segment Table, Transformation, Visibility

(**Date of Acceptance:** 09-07-2015; **Plagiarism Check Date:** 13-07-2015; **Peer Reviewed by Three editors blindly:**19-08-2015; **Reviewer's Comment send to author:** 25-09-2015; **Comment Incorporated and Revert by Author:** 28-09-2015; **Send for CRC:** 30-09-2015)

1. Introduction

Sometimes bioinformation may be composed of several pictures or items of information. A single image of bioinformation may contain several views of an object[1]. It may have a picture of the overall object and a close up of a particular component. As an example suppose we have image for DNA, RNA and Protein. So, some user only wants to display DNA structure as well some other user wants to display only RNA, someone wants Protein and DNA structure. There is also case when transformations will be applied to only RNA while another structure of DNA will remain as it is. So, we wish to apply them to only a portion of the picture either for DNA or for RNA or Protein or combination from group of these three structures[2].

So, it is required to organize display file to reflect this subpicture structure. For that the display file will be divided into segments where each segments corresponding to components of the overall display. For storing display file information of any part of bioinformation image there is need for some special hardware and software instructions. Then a set of attributes will be assigned to each segment. One of them is visibility. A visible segment will be displayed but an invisible segment will not be shown[3]. By varying the settings of the visibility attributes we can build a picture out of the selected subpictures. Consider the DNA, RNA and Protein structure. For example in first segment select DNA images in second segment select RNA images and for Protein images select third segment. Now by making the first and second segments visible and the third invisible only DNA and RNA will be displayed. While all display information may be present in the computer one can selectively show portion of it by designating which segments of the display files are to be executed.

Other attributes that can be associated with each segment is an image transformation. This will allow the independent scaling, rotating and translating of each segment. For the DNA, one can zoom while the effect of this zooming will not affect the RNA structure because the RNA structure will be stored in another segment of display file. It is also possible to shift either of them structure by means of an image transformation while leaving the other unchanged.

In this chapter segmentation of the display file is discussed and some algorithm suggested for create, close, rename and delete segments. Then implement display file attributes as visibility and image transformation[4].

2. Segment Table for DNA, RNA and Protein Structure

For implementing the concept of display file segment for different bioinformation related with DNA, RNA and Protein structure

*Author for correspondence

and their various types of images it is required to assign unique name so that they will uniquely identified. For changing visibility of a particular segment on and off attribute will be assigned to that segment and the on and off attribute does not affect other segment[5]. In this implementation it must be known that which display file instruction belongs to it. This may be determined by knowing where the display file introduction for the segment begins and how many of them there are. For each segment, we shall need some way of associating its display-file position information and its attributes information with its name. This concept can be implemented by creating a segment table. In this number of the segment will be used. For holding segment properties simple arrays will be used and the segment name will be used as unique identity and index into these arrays. In this one array containing display file starting locations. A second array will hold segment size information while a third will indicates visibility and other attributes will be stored in array.

The number 1 will refer to the first named segment of the table then 2 to the second named table entry and so on. This segment table has different entries for attributes. For holding the display files starting positions, for the segment size and for the attributes such as visibility and the image transformation parameter. If any user not to show segment number 3 he will set the corresponding entry in the visibility array to off. When anyone wants to show bioinformation[6] images then the segment table to determine which segment are visible. For each visible segment look up its starting point and size and pass this information along to our display file interpreter. The interpreter will therefore only interpret those segments which are visible.

Segment table can be illustrated from Table 1.

In Table 1 suppose there is display file image information for DNA is stored. So if visibility of segment 1 is on then it will be displayed and the other attribute like tx, ty etc. for image transformation is set then the effect of transformation will only affect to segment 1. There is no effect of visibility and transformation to other segments.

3. Illustration for Bioinformation

In Figure 1, segment 4 display file segments are there[7]. Every segment will store bioinformation for any one part of the entire

Table 1. Segment table

Segment Number	Start Position	Size	Visibility
1	1	15	On
2	16	30	Off
3	37	14	On
4	52	11	On

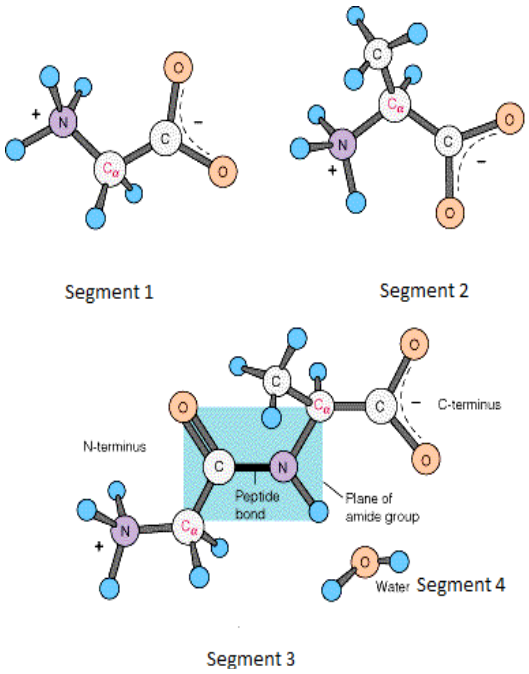

Figure 1. Concept of segment implemented for bioinformation.

image using special hardware and software routines. Now after defining the segment the image contained in that segment will be managed individually[8].

4. Creation of Segment

For creating a new segment it is required to insert the corresponding block of instructions in the display file and to create entry in the segment table[9]. A segment can start from the first available location in the display file. Image drawing commands will be the members of the created segment. A unique name will be given to every new created segment so it can be identified[10]. Suppose it noted that this is segment number 3 then all following image generation instructions would belong to segment 3. After creating segment 3 another segment can be created. Suppose it is 5. So, as per our requirements segment will be created.

5. Suggested Algorithm for Creation of Segment

Algorithm CREATE-SEGMENT(SEGMENT NAME) Instructions to create a segment[11].

Arguments SEGMENT-NAME the segment name.
Global
FREE the index of the next free display-file cell

```
                SEGMENT-START, SEGMENT-SIZE,
                VISIBILITY, ANGLE, SCALE-X, SCALE-Y,
                TRANSLATE-X, TRANSLATE-Y arrays that
                make up the segment table.
BEGIN
    SEGMENT-START[SEGMENT-NAME] <- FREE;
    SEGMENT-SIZE[SEGMENT-NAME] <- 0;
    VISIBILITY[SEGMENT-NAME] <- VISIBILITY[0];
    ANGLE[SEGMENT-NAME] <- ANGLE[0];
    SCALE-X[SEGMENT-NAME] <- SCALE-X[0];
    SCALE-Y[SEGMENT-NAME] <- SCALE-Y[0];
    TRANSLATE-X[SEGMENT-NAME]          <-
    TRANSLATE-X[0];
    TRANSLATE-Y[SEGMENT-NAME]          <-
    TRANSLATE-Y[0];
END;
    DF-X[FREE] <- X;
    DF-Y[FREE] <- Y;
    FREE <- FREE + 1;
END;
```

6. Deletion of Segment

If a segment is not needed then it can be deleted. The method of doing this depends upon the data structure used for the display file[12]. We have used arrays. Recovery of a block of storage in an array is both simple and straight forward, but it is not as efficient as some other storage techniques. What we shall do is take all display files instructions entered after the segment which we are deleting was closed, and move them up in the display file so that they lie on the top of the deleting segment. Thus we fill in the gaps left by the deleted blocks and recover an equivalent amount of storage[13] at the end of the display file.

7. Suggested Algorithm for Deletion of Segment

DELETE-SEGMENT(SEGMENT-NAME) User routine to delete a segment[14].

```
Arguments   SEGMENT-NAME the segment name.
Global      NOW-OPEN the segment currently open
            FREE the index of the next free display-file cell
            DF-OP, DF-X, DF-Y the display-file arrays.
            SEGMENT-START, SEGMENT-SIZE,
            VISIBILITY part of the segment table arrays.
Constant    NUMBER-OF-SEGMENTS size of the seg-
            ment table.
Local       GET the location of an instruction to be moved
            PUT the location to which an instruction
            should be moved
            SIZE the size of the deleted segment
            I a variable for stepping through the segment
            table.
BEGIN
    IF SEGMENT-NAME < 0 OR SEGMENT-NAME >
    NUMBER-OF-SEGMENTS THEN
            RETURN ERROR 'INVALID SEGMENT
            NAME';
    IF SEGMENT-NAME = NOW-OPEN AND
    SEGMENT-NAME <> 0 THEN
            RETURN ERROR 'SEGMENT STILL OPEN';
    IF SEGMENT-SIZE[SEGMENT-NAME] = 0 THEN
    RETURN;
    PUT <- SEGMENT-START[SEGMENT-NAME]
    SIZE <- SEGMENT-SIZE[SEGMENT-NAME];
    GET <- PUT + SIZE;
    shift the display-file elements
    WHILE GET < FREE DO
    BEGIN
            DF-OP[PUT] <- DF-OP[GET];
            DF-X[PUT] <- DF-X[GET];
            DF-Y[PUT] <- DF-Y[GET];
            PUT <- PUT + 1;
            GET <- GET + 1;
    END;
    recover the deleted storage
    FREE <- PUT;
    update the segment table
    FOR I = 0 TO NUMBER-OF-SEGMENTS DO
    IF SEGMENT-START[I] > SEGMENT-
    START[SEGMENT-NAME] THEN
            SEGMENT-START[I] <- SEGMENT-
            START[I] - SIZE;
    SEGMENT-SIZE[SEGMENT-NAME] <- 0;
    IF VISIBILITY[SEGMENT-NAME] THEN NEW-
    FRAME;
    RETURN;
END;
```

8. Visibility

Each segment is given a visibility attributes[15]. The segment's visibility is stored in an array as part of the segment table. We are therefore able to look up the value of each segment's visibility. By checking this array we can determine whether or not the segment should be displayed. We should give the user some method of changing the value of a segment's visibility[16], so that he can choose to show or not to show the segment. The following algorithm

does this, while freeing the user from any concern about the global variables and segment table representation. If the visibility is being turned[17] off, then a new-frame action is needed.

9. Conclusion

Display file segment is helpful for displaying image as per requirements of the user. Every part of the image can be easily managed. In this concept transformation can be applied easily. Processing is also fast because it only stores primitive command.

10. References

1. Attwood TK, Smith DP. Introduction to Bioinformatics. 1st ed. Pearson Education; 2001.
2. Gibas C, Jambeck P. Developing Bioinformatics Computer Skills. 1st ed. O'Relly and Associates; 2001.
3. Higgins D, Taylor W. Bioinformatics – Sequence, Structure and Databanks. 2nd ed. Oxford University Press; 2003.
4. Hearn D, Baker P. Computer Graphics. 2nd ed. Prentice Hall; 2000.
5. Harrington S. Computer Graphics – A Programming Approach. 2nd ed. McGraw Hill Book Company; 1987.
6. Tisdall J. Beginning Perl for Bioinformatics. 1st ed. O'Relly and Associates; 2001.
7. Nair AS, Sunitha P, Aswathi BL. Bioinspired Computing– the Evolving Scenario. CSI 2011; 35(9):6–8.
8. Govindan G, Goli B. WEKA – A powerful free software for implementing Bio – inspired Algorithms. CSI. 2011; 35(9):9–11.
9. Tripathi KK. Bioinformatics: The Foundation of Present and Future Biotechnology. Current Science. 2000; 79(5).
10. Kumar S, Tamura K, Nei M. Evolutionary Genetic Analysis and Sequence Alignment. Briefings in Bioinformatics. 2004; 5(2):150–63.
11. Mukhopadhyay A, Chattopadhyay A. Computer Graphics and Multimedia. 2nd ed. 2007.
12. Anderson G. Bioinformatics Tutorial rev. 9-2012, Bio 242 | Cellular and Molecular Biology. 2012. p. 1–10.
13. Newman W, Sproull R. Principles of Interactive Computer Graphics. 2nd ed. New York: McGraw-Hill; 1979.
14. Pavlidis T. Algorithms for Graphics and Image Processing. Rockville, Md.: Computer Science Press; 1982.
15. Rogers DF, Adams JA. Mathematical Elements for Computer Graphics. New York: McGraw-Hill; 1976.
16. Scott JE. Introduction to Interactive Computer Graphics. New York: Wiley Interscience; 1982.
17. Sherr S. Vide and Digital Electronics Displays: A user's Guide. Wiley; 1982.

Citation:
Chintankumar Dayalal Gohel
"Utilisation of Segment for Displaying of DNA, RNA and Protein Related Images",
Global Journal of Enterprise Information System, Vol. 7 | Issue 3 | July-September 2015 | www.gjeis.org

Conflict of Interest:
Author of a Paper had no conflict neither financially nor academically.