

Недобежкин Павел Владимирович

ПММ, 61 группа, 4 курс

Отчет по лабораторной работе №3

Задание 1.

Найти наилучшее локальное выравнивание между двумя строками нуклеотидных или аминокислотных последовательностей. Файлы для примера брать в соответствии с вариантом из Лабораторной работы 2 из базы данных GenBank.

Входные данные. Две строки V и W и матрица весов.

Выходные данные. Локальное выравнивание, определяемое подстроками строк

V и W, глобальное выравнивание которых определенное матрицей весов, является

наилучшим среди всех глобальных выравниваний всех подстрок V и W.

Код с примером

```
public class Main {
    public static void main(String[] args) {
        String s1 = "ACTGATTCA";
        String s2 = "ACGCATCA";
        int[][] matrix = { { 2, -3, -3, -3 }, { -3, 2, -3, -3 }, { -3, -3, 2, -3 }, { -3, -3, -3, 2 } };

        String[] res = NeedlemanWunschAlgorithm(s1, s2, matrix, -2);
        System.out.println(res[0]);
        System.out.println(res[1]);
    }

    static String nucleotidesToInd = "ATGC";

    public static String[] NeedlemanWunschAlgorithm(String s1, String s2, int[][] matrixOffFine, int spaceFine) {
        int[][] matrix = new int[s1.length() + 1][s2.length() + 1];
        matrix[0][0] = 0;
        for (int i = 1; i <= s2.length(); i++) {
            matrix[0][i] = matrix[0][i - 1] + spaceFine;
        }
        for (int i = 1; i <= s1.length(); i++) {
            matrix[i][0] = matrix[i - 1][0] + spaceFine;
        }

        for (int i = 1; i <= s1.length(); i++) {
            for (int j = 1; j <= s2.length(); j++) {
                matrix[i][j] = Math.max(Math.max(matrix[i - 1][j] + spaceFine, matrix[i][j - 1] + spaceFine),
                    matrix[i - 1][j - 1] +
                    matrixOffFine[nucleotidesToInd.indexOf(s1.charAt(i - 1))][nucleotidesToInd
                        .indexOf(s2.charAt(j - 1))]);
            }
        }
    }
}
```

```

    }

    int i = s1.length();
    int j = s2.length();
    String[] result = new String[2];
    result[0] = "";
    result[1] = "";

    while (i != 0 || j != 0) {
        if (matrix[i - 1][j - 1] > matrix[i - 1][j] && matrix[i - 1][j - 1] >
matrix[i][j - 1]) {
            result[0] = s1.charAt(i - 1) + result[0];
            result[1] = s2.charAt(j - 1) + result[1];
            i--;
            j--;
        } else if (matrix[i - 1][j] > matrix[i - 1][j - 1] && matrix[i -
1][j] > matrix[i][j - 1]) {
            result[0] = s1.charAt(i - 1) + result[0];
            result[1] = "_" + result[1];
            i--;
        } else {
            result[0] = "_" + result[0];
            result[1] = s2.charAt(j - 1) + result[1];
            j--;
        }
    }

    return result;
}
}

```

Результат выполнения примера

```

ACTG_ATTCA
AC_GCAT_CA

```

Результат выполнения собственного варианта со своей матрицей замен

```

CAGGAGCTGCT GT GTGG GGCCA GTCTA ATCGATGATG AG_TGG AT CCTCAGTGCAGCC CACTGCATCCT CTATCC ACC ATGG AACAAACTTCACCATT A A TGACATCTGTGTC GGCCT T GGCANACA TAACA G AGC TA AGTTTG AGAAGGGCACA G AGAAGATTG T GGCTATTGAT
G_A G AT_A ATTGTCCACCCAGTACACTGGAAG AGAACCTG A ACCG G GACATTG CTC TGCT GC AC ATGA GG AGGC C CATTAC TTTCAC AGA TGAGATTG ATCTG TCTGTC TAC C AACCCAGCAGT TG C TAAGACACT GATG TTG CTG GCTATA A A GGC CGTGTGACA
G G CTGGGGGACCT ATATGAGAC ATGAGTTTC CTC C CCAAGTC TC TACCCACA G TTC TC CAGCABAT CCAT CTAC C CAT CGTGG AAC A GATA TC TGACAGAC TCTA CCTCTA T CGGCAT CACTGAC AATA TGTCTGTGCTGGCTTCAA A CCAGA GGAC AGA A A
ACTG G TG AGCG TTGTG AAGGG AGAC CG GTGGTCCCT T TG TATGAGAGG CCGATGAGACCC TTGGTACCAT CCGCATG TGCTCTGGGG AGAG AT GTGA CAGGATGGGAAAT A TGGATTTCAC CC CATC TTTTC C G TATGAG ACGTGGATGAGAAAGT T ATTG ACAGA
ACAG G CG G CGATG A CG ATG ACTG ATTGTTA TTCT C ATTTTCTCTACATGCAAGGAA A CTGA TG TA ATA TIG GAATA AA CA T GGGT T CCT G ATCATG
CAGGAGCTC AT ATGTGGAGC AAG CA TCA TCAGTA CCG CTGGGT TC TCACCTGAGGCG ACTGCAT TTCTTA CCCACCT GGBA CAANACTACACCA CA GA A GACATCTGTGTC G AAT TGG AAA AGCTA C AGGAC CA A GTA CGAGA G ACACAGGAGAGATTCGAATGC T G
G AGCGGATCATCA TTC ACCCCAGTACAACTGGA GGGAGAACCTGGACAG G G ACAT CGC CCG ATC CAGCT GAAGCGAC C C ATT GGCTT CAACA ACT ACAT CCATCC CGTCTG CCT GC CCA CCAAGGAG AT CGTCCA GAC GTTGATGCT GAACAGAC A CA A AGGGC GTGTGTC
CG GCTGGGGGACCTGCAT GAGACCT GGAC C TCGGGGGGC C A G GC CCTTCCC C AGGT CCTGC AGCAG GTC AATCT GC C AATCGT GGA CCAAG AA A CCTGCA A AGCCTCA ACCA AAATC AAAGTCACT AGCAA CATGTTCTGTGCGAG TT ATA AACCAGATG A GCCA A CA
G AG G GAGCGCT G CGA GGGGGACAGTG G TGCTCC AT T CGTCATGAGAGTCC AGATGACAAACCGT GGTACCG GTCCGCAT CGTCTCTGGGG CGA GGGT TGTGATC GGGATGGCAA GT ATGGATTCTA TAAGC A CCT GC ACCGGA TG CGCCAG TGGATGATGA GAT CAT CGAGAA
G TG TG G G A GCTAG GA GTGA TGCA G CCA GCTTCATGCAT C CA CA G AAG AGGCAAAACA TATCTAGAA AT GTCTG AAAA A TACAGCCATAAAGGCTCATTC TTCGGGAACA GC

```

Найти наилучшее локальное выравнивание между двумя строками нуклеотидных или аминокислотных последовательностей. Файлы для примера брать в соответствии с вариантом из Лабораторной работы 2 из базы данных GenBank.

Входные данные. Две строки V и W и матрица весов.

Выходные данные. Локальное выравнивание, определяемое подстроками строк

V и W, глобальное выравнивание которых определенное матрицей весов, является

наилучшим среди всех глобальных выравниваний всех подстрок V и W.

Код с примером

```
public class Main {
    public static void main(String[] args) {
        String s1 = "ATGCATCCCATGAC";
        String s2 = "TCTATATCCGT";
        int[][] matrix = { { 2, -3, -3, -3 }, { -3, 2, -3, -3 }, { -3, -3, 2, -3 }, { -3, -3, -3, 2 } };

        String[] res = SmithWatermanAlgorithm(s1, s2, matrix, -2);
        System.out.println(res[0]);
        System.out.println(res[1]);
    }

    static String nucleotidesToInd = "ATGC";

    public static String[] SmithWatermanAlgorithm(String s1, String s2, int[][] matrixOfFine, int spaceFine) {
        int[][] matrix = new int[s1.length() + 1][s2.length() + 1];
        for (int i = 0; i <= s2.length(); i++) {
            matrix[0][i] = 0;
        }
        for (int i = 1; i <= s1.length(); i++) {
            matrix[i][0] = 0;
        }

        int iMax = 0, jMax = 0;

        for (int i = 1; i <= s1.length(); i++) {
            for (int j = 1; j <= s2.length(); j++) {
                matrix[i][j] = Math.max(Math.max(matrix[i - 1][j] + spaceFine, matrix[i][j - 1] + spaceFine), Math.max(
                    matrix[i - 1][j - 1] +
                    matrixOfFine[nucleotidesToInd.indexOf(s1.charAt(i - 1))][nucleotidesToInd
                        .indexOf(s2.charAt(j - 1))],
                    0));
                if (matrix[i][j] > matrix[iMax][jMax]) {
                    iMax = i;
                    jMax = j;
                }
            }
        }

        String[] result = new String[2];
        result[0] = "";
        result[1] = "";

        while (matrix[iMax][jMax] != 0) {
            if (matrix[iMax - 1][jMax - 1] >= matrix[iMax - 1][jMax]
                && matrix[iMax - 1][jMax - 1] >= matrix[iMax][jMax - 1]) {
                result[0] = s1.charAt(iMax - 1) + result[0];
            }
        }
    }
}
```

```

        result[1] = s2.charAt(jMax - 1) + result[1];
        iMax--;
        jMax--;
    } else if (matrix[iMax - 1][jMax] >= matrix[iMax - 1][jMax - 1]
        && matrix[iMax - 1][jMax] >= matrix[iMax][jMax - 1]) {
        result[0] = s1.charAt(iMax - 1) + result[0];
        result[1] = "_" + result[1];
        iMax--;
    } else {
        result[0] = "_" + result[0];
        result[1] = s2.charAt(jMax - 1) + result[1];
        jMax--;
    }
}
return result;
}
}

```

Результат выполнение примера

```

ATCC
ATCC

```

Результат выполнения задания

```

CAGGAGCTGCT GTGTGG GGGCA G T CTAATCAGTGAATG AG TGG AT CCTCACTGAGCC CACTGATCTCT CTATCC ACC ATGG AACAAAACCTTCCCAATTA A TGCATCTGTGTC GGCCT T GGCAMA CA TACAGG AGC TA AGTTTG AGA ABBCCA CAG AGAAGATTC T GCTATTGATG AG
AT A ATGTGTCACCCGANGTACACTGGGAGG AGAATCTG A AGCG G GACATTG CTC TGCT GC AC ATGA GG ABBG C GATTAC TTTCAC AGA TGAGATTG ATCTG TCTGTC TAC C AACCAAGCAGGTTG C TANGACCT GATG TTTC CTG GGTATANAGGC CGTGTGACAG G CTGGGGGA
CCT ATATGAGAC ATGAGTTC CTC C CAAGTC TC TACCCACAG TTC TC CAGCAGATCCAT CTAC C CAT CGTTG AAC A GGATA TC TGAGAGAGC TCTA CCTCTA T CGGA T CACTGAC AATATGTTCTGTGCTGCTTCAA A CCAGA G G AA CAGAAAATG G TG AGCG TTG
TG AAGGGG ACAG CG GTGTCCT T TG TCATGAAGAG CCCAGATGACACCG TTGTACCGATCGGATTG TGTCCTGGGGAG AAGG AT GTGA CAGGGATGGGAATATGATTCACACC CATC TTTTC C G TATGAG ACGGTGGATGAAGAA AGTTATTG ACAAACAG G CG G CGATG A CG A
TG ACTG ATTGTTATTC TTCT C ATTTTCTCTACATGCAAG G AAA CTGA TG TA ATA T TG GAAATA A ACA T GGGT T CCT G ATCATG
CAGGAGCTC ATATGTGGAGC AAGCATCA TCAGTGA CCG CTGGGT TC TCACTGCAGCGC ACTGCAT TTTCTA CCCACCTT GGGG CAAAACTACACCACA GA A GACATCCTGGTGC G AAT TGG AAAACACTA CAGGAC CA A GTA CGAGAGAC AACAGGAGAAGATTCGAATGC T G GAGCGG
ATCATCA TTC ACCCCANGTACACTGGA GGGAGAACCTGGACAG G G ACAT CGC CCG ATC CAGCT GAGCGAC C C ATT GGGTT CACCA ACT ACAT CCATCC CGTCTG CCT GC CCA CCAAGGAGAT GTTCCA GAC GTTGATGCT GANCAGAC A CAAAGGC GTGTGTC CG GCTGGGGGA
CTGCAAT GAGACCT GAGC C TGGGGGGG C A G GC CCTTCCC CAGGT CCTGC AGCAGTC ATCT GC C AATCGT GGA CCAGG AA A CTTGCA A AGCTTCA ACCA AATC AAGTCACT AGCACAATGTTCTGTGCAG TT ATA AACCAATGAGCCAAACAGA G G G GAGCCT G
CGA GGGGAGACATG G TGCTCC ATT T CGTCATGAGAGATCC AGATGACAAKCGCT GGTACAGATGGGAT CGTCTCTGGG CCA GGGT TGTGATC GGGATGCCAGTATGGATTCTATACGC A CCT GCA CCGA TG CGCCAG TGATGATGAAGA TCAT CGAGAA G TG TG G G A GCTAG GA
GTGGA TGCA GCGA GCTTCATGCAT C CA CA G AAGGAGCAAAACA TATCTAGAA ATGTCTG AAAA A TA CAGGCCAATAAAGCCTCATTC TTCGGGAACA G
PS D:\code\labs\B1oInf4>

```