### 用动态规划方法解钢条切割问题

### Hushrush.2021.04.24

1. 动态规划复习

动态规划算法（Dynamic Programming，DP）：

* 思想是将待求解问题分成若干较小的子问题（类似于分治法），先求解子问题，再从这些子问题得到原问题的解。
* 特点是

1. 适合动态规划方法的问题分解得到的子问题往往是不独立的，而分治法的子问题是独立的。
2. 动态规划方法适合求解**最优化**问题。

* **做法是用一张表（备忘录）记录所有子问题的答案，避免重复求解。**
* **一般步骤**

1. 问题建模，**写出目标函数**；
2. 划分子问题；
3. 找出依赖关系，**写出递推方程**；
4. 验证最优子结构；
5. 构造最优解（必要时）。

* 实例详见**《动态规划复习》**一文

1. 钢条切割问题

设有一个长度为 L 的钢条，在钢条上标有 n 个位置点（p1,p2,…,pn）。现在需要按钢条上标注的位置将钢条切割为 n+1 段，假定每次切割所需要的花费与所切割的钢条长度成正比。请编写⼀个算法，能够确定⼀个切割方案，使切割的总花费最小。

输入格式：

20 个测试用例， 每个测试用例两行。第一行输入钢条长度 L 和位置点个数 n。第二行输入 n 个切割点位置 p1,p2,…,pn（每一个切割点位置是该切割点距钢条起始点的距离， 且切割点的位置乱序）

例如：

7 4

1 3 4 5

输出格式：

每个测试用例输出一个切割最小的花费和

上述的输出应为

16

数据规模：

⚫ 2 ≤ L ≤ 105

⚫ 1 ≤ n ≤ 200

⚫ 1 ≤ 𝑝𝑖 < 𝐿

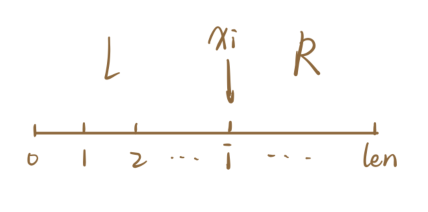


图2.1 钢条示意

如图2.1为一**钢条**。0、1、2、…、为坐标，设为一**切割点**且位于坐标处。切断后，钢条分为左右两段**子钢条。**现设钢条的左右端点为，则左右两子钢条可以记作(),()。按照这种记法，原钢条可以记为()。

要切下一段钢条需要一次或两次切割：切掉端点处的钢条需要一次；切掉中间的钢条需要两次[[1]](#footnote-2)。现按照**三元组**的记法来确定一段**切割**：记三元组()为切割,。其中为切割点，则一个三元组中有正整数个切割点。以二元数组（矩阵）的形式建立**备忘录**记录任意三元组最小**总花费**[[2]](#footnote-3)，如图2.2所示。

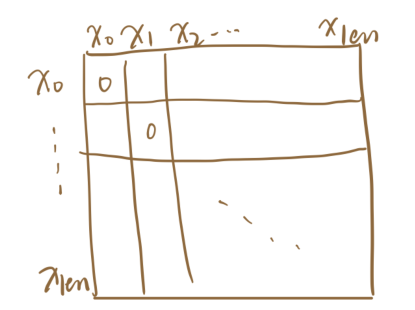


图2.2 矩阵存储示意

总花费是指将该三元组切割到不可再分的所有花费。由于给定三元组的端点就确定了本次切割花费，故定义总花费的概念。钢条的花费与长度成正比，则不妨取二者相等。由此有某一三元组的总花费

由此递推式可以求出元钢条的最小花费。

现初步设计程序：

1. 设钢条长度为barlen。经实际编程后发现对于钢条只需要知道长度。
2. 可用一数组x存储切割点。 如图2.4所示，设切割点共有xnum个，则x长度应为xnum+2[[3]](#footnote-4)。

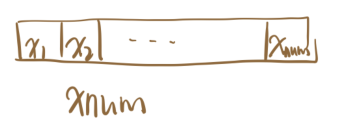


图2.4 切割点存储示意

1. 输入后动态生成两数组，先初始化再逐渐优化。

代码如下：

#include"iostream"

#include"algorithm"

#define INF 100000000

using namespace std;

int num; int barlenn;

int main()

{

    int lowestCost(int m, int n, int\* &x, int xlen, int\*\* &memo);

    int\* x;//切割点数组

    int\*\* memo;//备忘录矩阵

    int barnum, xnum;

    cin >> barnum >> xnum;

    int xlen = xnum + 2;

    num = xlen;

    int barlen = barnum + 1;//应有长度

    x = new int[xlen];

    x[0] = 0; x[xlen - 1] = barnum;

    for (int i = 1; i < xlen - 1; i++)

    {

        cin >> x[i];

    }

    //切割点排序

    sort(x, x + xlen);//注意排序函数用法

    //初始化备忘录

    memo = new int\*[xlen];

    for (int i = 0; i < xlen; i++)

        memo[i] = new int[xlen];

    for (int i = 0; i < xlen; i++)

    {

        for (int j = 0; j < xlen; j++)

            memo[i][j] = INF;

    }

    int result;

    result = lowestCost(0, xlen - 1, x, xlen, memo);

    cout << result << endl;

    //释放指针数组

    if (x)

        delete[] x;

    if (memo)

    {

        for (int i = 0; i < xnum; i++)

        {

            if (memo[i])

                delete[] memo[i];

        }

        delete[] memo;

    }

    return 0;

}

int lowestCost(int m, int n, int\* &x, int xlen, int\*\* &memo)

{

    for (int i = 0; i < xlen; i++)

    {

        memo[i][i] = 0;

        if (i + 1 < xlen)

        {

            memo[i][i+1] = 0;

        }

        if (i + 2 < xlen)

        {

            memo[i][i+2] = x[i + 2] - x[i];

        }

    }

    int temp;

    //应采用步长

    for (int step = 3; step <= x[xlen - 1]; step++)//注意边界

    {

        for (int m = 0; m < xlen; m++)

        {

            if (m + step <= xlen - 1)

            {

                for (int i = m + 1; i < m + step; i++)

                {

                    temp = memo[m][i] + memo[i][m + step] + x[m + step] - x[m];

                    if (temp < memo[m][m + step])

                    {

                        memo[m][m + step] = temp;

                    }

                }

            }

        }

    }

    return memo[m][n];

}

值得注意的是：

1. 输入的xnum要加2，因为把端点也看做切割点。
2. memo只需要存储切割点即可，而不需存储其他点。
3. lowestCost函数中应采用步长而不是循环每一行每一列。这是因为大步长的结果需要用到小步长的结果。

其他的测试用例如下，全部通过。

输入：

|  |
| --- |
| 2 1  1  7 3  1 4 2  7 6  3 5 4 1 2 6  50 10  40 46 17 16 29 24 34 6 1 20  101 20  49 56 30 32 98 74 7 97 27 4 17 72 38 31 64 66 11 39 10 12  239 50  17 151 215 57 173 166 55 167 132 230 45 184 106 236 150 231 211 182 98 153 159 152 32 195 207 134 112 237 24 63 12 139 198 48 10 192 105 21 140 201 222 104 228 171 90 81 189 227 212 221  500 100  237 405 236 89 226 167 161 400 330 334 303 353 6 52 361 374 271 453 200 159 422 7 415 34 102 125 148 100 294 393 124 253 8 475 476 233 256 299 15 284 107 28 60 191 150 442 248 197 332 240 337 14 320 119 285 483 497 78 123 53 80 235 4 9 241 238 121 460 117 219 427 201 486 470 348 254 340 350 217 215 311 482 218 428 491 106 126 295 12 109 192 403 351 203 385 490 367 244 378 170  888 150  170 805 131 739 190 501 532 842 529 758 722 875 258 779 460 788 826 42 234 263 553 456 830 39 54 217 273 728 474 300 317 809 100 76 26 272 449 391 172 34 414 885 649 651 329 281 193 835 490 314 814 681 99 32 103 507 806 832 398 650 654 801 69 247 358 171 756 824 30 16 720 521 458 613 411 182 614 572 57 840 22 571 29 7 615 330 745 426 52 118 164 363 280 389 716 611 618 357 248 440 283 383 736 847 424 70 211 594 5 341 155 691 161 185 789 561 15 468 697 851 466 45 560 509 476 539 338 550 71 270 526 787 372 589 231 700 403 244 644 493 726 544 189 540 765 133 815 362 639 139  1000 200  740 585 582 794 15 354 404 364 538 812 307 348 600 618 277 296 226 596 942 852 408 945 785 721 671 66 703 331 647 199 808 590 980 416 656 99 286 753 326 391 39 58 458 355 715 524 688 822 337 621 675 399 765 9 906 905 820 901 411 800 469 650 947 462 5 489 550 432 324 370 610 70 288 722 770 562 407 400 230 789 97 368 380 774 106 669 520 34 67 576 52 685 452 691 142 733 220 724 243 131 13 315 137 373 659 511 10 963 739 490 246 181 345 249 463 358 360 38 627 810 2 759 992 464 297 50 415 94 897 421 441 868 132 801 457 210 616 418 983 741 805 932 96 92 395 116 697 386 55 588 501 839 981 42 895 639 444 480 273 896 294 712 385 74 93 268 848 679 754 256 567 35 884 861 622 313 147 430 310 282 918 704 222 658 494 583 27 488 409 956 587 281 468 930 682 874 575 941 826 68  2000 77  1166 299 1574 904 274 380 1202 1541 819 951 794 1183 878 1914 1411 1910 765 133 917 175 1185 1233 1916 1172 279 1437 1504 954 173 415 502 1597 1044 1800 1029 1939 81 1976 1161 25 816 659 354 267 1302 103 615 1588 932 1198 1309 1950 524 405 218 556 735 1072 402 542 351 1556 454 1841 935 48 1446 1958 411 1940 1640 619 544 1252 84 211 782  3000 45  840 756 2109 1643 93 852 129 282 2770 2860 1374 721 1046 815 2479 2961 2134 2185 1973 2154 650 747 2732 2590 678 784 442 120 2064 582 1787 2843 255 2688 134 701 1847 2072 1470 335 2559 2577 2782 1602 1970  4000 78  3242 1738 3382 778 3682 1236 478 2924 1662 3291 50 1947 581 113 2521 2706 2159 110 2268 736 2572 356 1372 2814 735 2517 1533 703 2199 91 2149 111 2469 2493 186 3387 3163 693 1780 1742 203 194 2867 3913 2894 3801 3054 3919 1918 3140 307 1701 685 211 2130 1638 3499 2719 1420 3320 632 2274 301 2416 946 699 3114 3324 3722 2191 975 2460 3018 1657 3917 354 923 192  5000 34  36 3361 4625 2684 3775 3005 2905 1274 927 1219 1781 806 4219 4637 3683 3957 3483 574 1241 4957 4639 559 4240 3346 3056 1561 1997 995 1214 3450 1753 1867 1889 4126  6000 134  877 327 5275 5595 5440 2334 5466 3214 29 1289 3794 40 1169 3495 1090 369 376 2626 2603 4064 5571 3776 1368 5704 4540 5980 5582 657 2533 2213 1116 4584 4245 1418 1952 3410 3944 1518 4757 1153 4244 1706 3793 2092 1089 3928 2773 4147 4188 91 3175 5170 4400 5346 1532 5857 4508 124 1954 3873 4503 3445 5638 3350 2638 2445 495 187 751 827 1712 3570 3750 5124 484 3419 3102 961 4904 2444 2588 3810 1128 4566 3578 4065 2189 4981 5334 1580 3939 2978 5871 1270 260 2590 4988 5238 3841 2646 1625 4172 3244 578 2863 4461 4937 127 1857 4613 3905 5633 899 134 4650 5297 1616 3091 4457 732 2233 4681 4416 2685 5105 1751 3989 3059 23 4683 2721 2218 1118 3747  7000 98  5149 6699 286 2647 5008 6454 5055 222 4766 2467 3796 581 1184 3535 3807 2872 4312 974 36 5775 218 6404 3433 5632 1224 4178 6790 4600 4259 383 2807 524 5330 2905 4454 6013 4304 607 3937 3941 3901 3361 2692 6452 976 4794 369 1838 6467 3873 1703 3119 2056 1504 1878 2982 6559 6478 6262 4847 3470 4102 6949 6581 3054 5703 180 1047 5970 1774 2935 654 3065 5844 2053 6163 5982 1126 3979 5006 5015 611 2758 817 4269 2089 1356 3156 2770 4268 6220 4870 176 652 1114 5337 3791 1164  8000 56  4158 1139 2424 2879 5856 7684 5782 7904 3864 7847 115 7424 3846 2497 6223 3351 3198 1154 1420 819 2107 5744 3374 2308 4035 5719 1685 2652 7879 3107 5925 2543 6253 4822 5681 7548 6719 6342 3331 4815 2773 1613 5083 1307 7384 5377 5622 6598 2273 258 4708 3909 3175 5205 3435 2766  9000 150  1224 4418 7970 465 1087 2885 1601 3351 559 5024 3788 209 5082 8499 1645 8559 2825 8765 6757 7644 850 6980 3212 1056 3604 4735 8947 7166 6111 7553 3224 327 1494 3034 6181 1782 7662 2368 3739 4114 4976 733 1469 2997 2159 3530 6438 1704 481 3542 5987 1907 5287 3170 8004 4657 4815 5927 2467 5667 3931 6991 374 6563 3462 3201 8094 5894 1157 5858 1886 3306 1903 2013 3629 7280 7464 7691 5568 5460 5592 8985 5887 6302 2933 5843 7973 421 1281 5723 344 1213 1535 4569 2335 176 3075 4422 3284 6065 1678 7735 2849 3863 6826 1471 5541 819 3758 712 1968 217 4124 6801 2952 943 6773 4187 658 4943 5201 1858 334 6673 757 8470 3236 4375 8569 8684 7626 7847 4194 4111 1807 8353 7914 7958 1324 3731 4689 5794 6966 1 288 8108 6985 6134 7112 8468  10000 200  1011 4475 3983 9604 8428 5306 7991 4353 9962 2928 3247 4981 6330 6619 7993 2911 6938 8782 6026 762 2953 6187 6908 5421 6735 6021 4286 350 3077 9511 5112 9042 1398 3329 4957 9955 8792 4901 3109 3123 6375 5601 9964 7237 2812 1948 6822 4947 6662 9039 9655 5492 9781 7243 3174 3752 4022 4120 5866 6467 2067 7119 1229 2208 4673 3043 1133 9050 639 4421 2706 895 7203 6160 2680 6027 171 7030 3985 2743 9651 4154 694 112 6668 1469 5586 507 5411 6276 7941 9995 7052 9755 9195 4922 8180 2085 5012 2103 7413 2235 2733 946 2615 7046 7871 7752 9278 6607 7039 1569 6610 6212 9127 6377 348 1471 4545 7514 3217 2685 5281 736 1694 4978 5641 6771 6151 816 5551 7059 1586 7756 4830 3188 1062 3882 6103 3810 9709 5209 9363 7690 5315 8721 5436 7910 2074 695 4810 2282 2162 7582 3984 8823 3363 5555 6036 6563 9425 3955 5534 7029 2747 8492 366 6009 2315 4625 4377 5120 37 3303 7722 1875 3892 5059 6698 5227 7101 9597 7408 1972 3001 2752 4061 1573 5365 2988 4346 1331 6448 2723 5170 8802 2640 1146 5503 9668  100000 34  48645 53022 25144 1549 43076 22249 50458 8457 70957 52470 49575 98611 29813 11312 69605 30757 34667 23500 53596 5328 32899 13411 14239 43528 49114 73405 22469 86096 29253 301 49490 31655 79695 29324  100000 200  83611 21863 6372 55199 5688 83462 11940 28000 14379 38678 37812 93018 81348 36466 60659 68228 6956 61100 14587 12617 74065 44533 9821 83996 36549 69521 15511 3360 48815 16830 31932 68806 95242 48014 53807 10266 54763 94714 49096 53841 26690 89439 58002 24747 57473 52629 72124 1655 43245 19747 48933 22414 7903 50752 79517 77335 31444 2675 83821 89577 31156 9091 88496 66703 95335 83643 3446 48530 99083 40556 9304 21311 94706 5467 43780 38240 58444 46596 969 53584 75897 8754 14210 8095 72278 80702 43532 68646 5451 57836 68288 71628 29678 36036 43946 59433 21558 21599 27050 21245 77450 87582 37006 3033 35656 15768 91397 83129 93881 44999 30755 71238 5753 79084 68843 44958 79401 98200 8828 78792 38210 23217 81814 84173 43886 69101 60762 92773 20208 39612 50341 28586 73575 27282 44663 26020 30411 14235 21010 98530 75517 5326 10320 57050 89801 45129 17767 49501 59609 35378 60855 74094 90632 46471 52191 94689 86794 32744 71100 8328 30155 36304 99965 19622 53610 33291 93587 29592 2706 68909 82610 76270 84841 29575 47373 49632 95039 63662 91957 58130 98829 42473 37142 82739 20535 13013 5272 26754 19022 16229 23204 46274 75481 79674 63985 40474 62188 55901 24052 245 |

输出：

|  |
| --- |
| 2  13  20  168  407  1272  3122  6126  7308  11632  15047  23217  23593  40383  43278  43035  61827  71344  447088  721592 |

1. 对于中间钢条，切掉一次后也变成了端点钢条。 [↑](#footnote-ref-2)
2. 本问题不需要构造解。 [↑](#footnote-ref-3)
3. 两端点视为切割点。 [↑](#footnote-ref-4)