动态规划选讲

重庆南开信竞

重庆南开中学信息学竞赛教练组

数位DP选讲

重庆南开信竞

例0 51nod1009 数字1的数量

在 10 进制下，写下从 1 到 n 的所有整数，问其中数字 1 共出现了多少次。

$1 \le n \le 10^9$

- 小学奥数——页码问题
- 分段计算，例： $n = 400$
  - 1~99：20次（个位10次，十位10次）
  - 100~199：120次（个位10次，十位10次，百位100次）
  - 200~299：20次
  - 300~399：20次
  - 400：0次

重庆南开信竞

例0 51nod1009 数字1的数量

- 很多段答案一致，它们有哪些相同的地方？
  - 百位不含 1
  - 个位和十位的组成都是 0~99

- 重复子问题，动态规划
- 怎么表示状态？要记录哪些信息？
  - 未确定数位的长度
  - 已经确定的数位中包含的 1 的数量

重庆南开信竞

例0 51nod1009 数字1的数量

- $dp[i][j]$ 表示：未确定数位（低位）还有 i 位，已经确定的数位（高位）中含有 j 个 1，构成的所有数字中 1 的个数

- 转移很简单，只需要枚举未确定位中的最高位
  - $dp[i][j] = 9dp[i-1][j] + dp[i-1][j+1]$
- 边界条件也很明确
  - $dp[0][j] = j$

- 在题目条件下，两维的大小均不超过10

重庆南开信竞

例0 51nod1009 数字1的数量

- 最后考虑如何利用求得的 dp 数组计算答案

- 将 1~n 分为多段求解
- 以 $n = 223$ 为例
  - 0~99： $dp[2][0]$
  - 100~199： $dp[2][1]$
  - 200~209： $dp[1][0]$
  - 210~219： $dp[1][1]$
  - 220、222、223 ： $dp[0][0]$
  - 221： $dp[0][1]$

重庆南开信竞

最多分为 $10logn$ 段

例0 51nod1009 数字1的数量

```
int len = 0;
while (n > 0) {
    num[++len] = n % 10;
    n /= 10;
}
int ans = 0, cnt1 = 0;
for (int i = len; i > 0; --i) {
    for (int d = 0; d < num[i]; ++d) {
        ans += dp[i - 1][cnt1 + (d == 1)];
    }
    cnt1 += (num[i] == 1);
}
ans += cnt1;
```

重庆南开信竞

- 先预处理好所有的 dp 值，并将 n 的各位数字求出
- 分段计算，同时统计已确定的高位中 1 的数量
- 最终还要统计 n 的答案（也可以在最开始将 n 加 1）
- 复杂度 $O(10logn)$

数位DP概述

- 通过例 0，我们已经简单感受了数位 dp

- 数位 dp 是用于求解 $0 \sim n$ 的整数中，满足一定条件的数的个数（或其它特征）这类问题的动态规划算法。这里的条件往往与数位有关。

- 数位 dp 的题大部分具有明显的特征，难点在于状态的设计

重庆南开信竞

数位DP的记忆化搜索写法

- 刚才介绍的是递推形式的数位 dp，只使用循环，常数较小
- 在状态和转移更为复杂的情况下，递归形式的记忆化搜索更加直观、好写，在时限不是很紧的情况下更推荐使用递归写法
- 使用记忆化搜索，数位 dp 就是**套模板**

重庆南开信竞

数位DP的记忆化搜索模板

```
int dfs(int pos, int st, bool limit) {
    if (pos == 0) return func(st);
    int& ans = dp[pos][st];
    if (!limit && ~ans) return ans;
    int up = limit ? num[pos] : 9;
    int ret = 0;
    for (int i = 0; i <= up; ++i) {
        ret += dfs(pos - 1, transfer(st, i), limit && i == up);
    }
    return limit ? ret : ans = ret;
}
```

重庆南开信竞

数位DP的记忆化搜索注意点

- 使用前初始化 dp 数组为-1，若-1可能成为常规的dp值，可以使用一个不可能出现的数值，或者额外开一个vis数组

- $limit$ 也可以再开一维存储，但有效期只有本组数据，因此一般情况下没有必要

重庆南开信竞

例0 记忆化搜索写法

```cpp
int dfs(int pos, int cnt, int limit) {
    if (pos == 0) return cnt;
    int& ans = dp[pos][cnt];
    if (!limit && ~ans) return ans;
    int up = limit ? num[pos] : 9, ret = 0;
    for (int i = 0; i <= up; ++i) {
        ret += dfs(pos - 1, cnt + (i == 1), limit && i == up);
    }
    return limit ? ret : ans = ret;
}
```

```cpp
memset(dp, -1, sizeof(dp));
printf("%d\n", dfs(len, 0, 1));
```

重庆南开信竞

例1 HDU 3555 Bomb

反恐精英在废墟中发现了一枚炸弹。这是经过恐怖分子改进了的定时炸弹，在定时器从 1 到 n 计时的过程中，如果当前时刻含有子串“49”，则爆炸威力将增加一个点。问最后爆炸的时候威力为多少。

$1 \le T \le 10^4$

$1 \le n < 2^{63}$

重庆南开信竞

- 多组数据、数据范围、题目描述都在明示数位 dp
- 尝试套模板?

例1 HDU 3555 Bomb

- 套板前先设计好状态

- 回忆单模式匹配（kmp）
  - 状态为主串的后缀与模式串前缀的最大匹配长度

- 该题中，匹配情况（状态）只有3种
  - 0：前面未出现“49”，前一位不是“4”
  - 1：前面未出现“49”，前一位是“4”
  - 2：前面已经出现了“49”

重庆南开信竞

例1 HDU 3555 Bomb

- 快乐套板

```
long long dfs(int pos, int st, bool limit) {
    if (pos == 0) return st == 2;
    long long& ans = dp[pos][st];
    if (!limit && ~ans) return ans;
    int up = limit ? num[pos] : 9;
    long long ret = 0;
    for (int i = 0; i <= up; ++i) {
        if (st == 2 || st == 1 && i == 9) ret += dfs(pos - 1, 2, limit && i == up);
        else if (i == 4) ret += dfs(pos - 1, 1, limit && i == up);
        else ret += dfs(pos - 1, 0, limit && i == up);
    }
    return limit ? ret : ans = ret;
}
```

重庆南开信竞

例2 51nod 1042 数字0~9的数量

在 10 进制下，写下从 a 到 b 的所有整数，问其中数字 0~9 分别出现了多少次。

$1 \leq a \leq b \leq 10^{18}$

- 常见套路：区间问题转化为两个前缀问题
  - 本题几乎就转化成例0了

- 一通套板，发现过不了样例
  - 数字 0 需要特殊处理

重庆南开信竞

例2 51nod 1042 数字0~9的数量

- 在 $n = 223$ 这个例子中，考虑 0~99 这段

- 该段使用的是 $dp[2][1]$ 进行计数，把百位的0（前导0）也统计进去了
- 此外，0~9 这些数也都统计了十位上的前导 0

- 因此，包含前导 0 的段需要特殊处理，处理方法同样是分段
  - 0~9
  - 10~99
  - 100~999

重庆南开信竞

数位DP记忆化写法中前导0的处理

- 再增加一个 $lead$ 标记即可

```cpp
long long dfs(int pos, int cnt, int limit, int lead) {
    if (pos == 0) return cnt;
    long long& ans = dp[pos][cnt];
    if (!limit && !lead && ~ans) return ans;
    int up = limit ? num[pos] : 9;
    long long ret = 0;
    for (int i = 0; i <= up; ++i) {
        ret += dfs(pos - 1, cnt + (!lead && i == 0), limit && i == up, lead && i == 0);
    }
    return limit || lead ? ret : ans = ret;
}
```

重庆南开信竞

- 前导0并不一定会影响答案，不影响时可不处理

例3 HDU 4507 吉哥系列故事——恨7不成妻

如果一个整数符合下面 3 个条件之一，那么我们就说这个整数与 7 有关

1、整数中某一位是 7；

2、整数的每一位加起来的和是 7 的整数倍；

3、这个整数是 7 的整数倍

求 $[L,R]$ 内与 7 无关的数字的平方和 模 10^9+7。

$1 \le T \le 10^4$

$1 \le L \le R \le 10^{18}$

- 前导0影响答案吗?　　不影响！

重庆南开信竞

例3 HDU 4507 吉哥系列故事——恨7不成妻

- 状态要记录哪些?
  - 是否含“7”
  - 数位和模 7 的值
  - 整个数模 7 的值

重庆南开信竞

```
for (int i = 0; i <= up; ++i) {
    ret += dfs(pos - 1, st1||i==7, (st2+i)%7, (st3*10+i)%7, limit&&i==up);
}
```

- 求数字的个数就做完了
- 求平方和有点跳跃，先考虑怎么求一次方和

例3 HDU 4507 吉哥系列故事——恨7不成妻

- $dp[i][st]$ 表示还剩 i 位未确定，已确定位的状态为 st ，能够组成的与 7 无关的数的个数，这部分套板就能轻松求出

- $sum[i][st]$ 表示还剩 i 位未确定，已确定位的状态为 st ，能够组成的与 7 无关的数的和
  - 这个状态表示能够进行递推吗？
  - 已确定位对于计算和的贡献无法从状态 st 中直接得到
- $sum[i][st]$ 表示还剩 i 位未确定，已确定位的状态为 st ，能够组成的与 7 无关的数中后 i 位数的和

重庆南开信竞

例3 HDU 4507 吉哥系列故事——恨7不成妻

- 此时状态转移只需要考虑新确定的一位的贡献，可使用 权值 $*$ 次数 计算

- 将整个数拆分为 数位 $*$ 位权值 的形式：$\overline{a_n a_{n-1} \ldots a_1} = \sum a_i * 10^{i-1}$

- $sum[i][st] = \sum_{j=0}^{9} sum[i-1][transfer(st,j)] + dp[i-1][transfer(st,j)] * j * 10^{i-1}$

- 再来考虑平方和
  - 常见套路：$(\sum a_i)^2 = \sum a_i^2 + 2\sum\sum a_i a_j$

重庆南开信竞

例3 HDU 4507 吉哥系列故事——恨7不成妻

- 设 $S_n=\sum_{i=1}^{n}a_i$
- $S_{n+1}^2=(S_n+a_{n+1})^2=S_n^2+a_{n+1}^2+2S_na_{n+1}$
- 因此，新确定一个数位后，平方和可以进行递推

- $squ[i][st]$ 表示还剩 i 位未确定，已确定位的状态为 st ，能够组成的与 7 无关的数中后 i 位数的平方和

- $$squ[i][st]=\sum_{j=0}^{9}\begin{array}{l} squ[i-1][transfer(st,j)]+ \\ dp[i-1][transfer(st,j)]*\left(j*10^{i-1}\right)^2+ \\ 2*sum[i-1][transfer(st,j)]*j*10^{i-1} \end{array}$$

重庆南开信竞

例3 HDU 4507 吉哥系列故事——恨7不成妻

- 在实现上，可以将 dp 值设为结构体，同时维护个数、一次方和与平方和
- 本题不难，但细节较多
- 暂不提供完整代码，请大家静下心来思考与理解，独立完成代码编写

重庆南开信竞

例4 51nod 1232 完美数

如果一个数能够被组成它的各个非 0 数字整除，则称它是完美数。例如：1~9 都是完美数，10、11、12、101 都是完美数，但是 13 就不是完美数（因为 13 不能被数字 3 整除）。

求 $[L, R]$ 内共有多少完美数。

$1 \leq T \leq 10^4$

$1 \leq L \leq R \leq 10^{18}$

重庆南开信竞

- 是否有 0 不影响答案，不用处理前导 0
- 状压?

例4 51nod 1232 完美数

- 考虑 $2^8 = 256$ 状压数字 2~9 是否出现

- 怎么判断整除?
  - 存储模 2~9 的值，$9! = 362880$，太多了
  - 有些模值可以由其它推出，例如 模2为0 可以转化为 模4为0或2
  - $8 \Rightarrow 2$、4 ， $9 \Rightarrow 3$ ， 8、$9 \Rightarrow 6$
  - 状态减少为 $5 * 7 * 8 * 9 = 2520$

重庆南开信竞

例4 51nod 1232 完美数

- 此时状态数为 $256 * 2520 * 18 = 11{,}612{,}160$，还有优化的空间

- 回忆扩展中国剩余定理
- 同时被多个数整除 $\Rightarrow$ 被这些数的 lcm 整除
- 在 $2^8 = 256$ 个数字组合中，只有 48 种 lcm 值，状态从 256 减少到 48
- 与此同时，我们也不用分别存储模为 5、7、8、9 的值，直接以所有数的 lcm 值 2520 为模即可（状态并未减少，仅优化写法)

- 数位 dp 的关键还是在于状态的分析和设计

重庆南开信竞

例5 51nod 1623 完美消除

消除操作：如果一个数的一段连续数位上的数字都相等（假设等于 x），那么可以选择一个 $d(1 \le d \le x)$，将这些位上的数字同时减去 d 。

最小操作数：将一个数变为 0 所需的最少的消除操作次数。

例如：$131131 \Rightarrow 111131 \Rightarrow 111111 \Rightarrow 0$，因此 131131 的最小操作数为 3

求 $[L, R]$ 中最小操作数为 k 的数的个数。

$1 \le L \le R \le 10^{18}$

$1 \le k \le 18$

- 看起来很像数位 dp，但完全不知道怎么设计状态

重庆南开信竞

例5 51nod 1623 完美消除

- 先考虑：给定一个序列，怎么消除可以使得操作次数最少

- 显然可以先把数值相同的一段长度缩成 1
- 操作次数的上界是缩减后序列的长度，即每一段消1次

- 贪心：每次选择数值最大的一段，将其消为左右两端中较大的一段的值
- $25342 \Rightarrow 23342 \Rightarrow 23332 \Rightarrow 22222 \Rightarrow 0$

重庆南开信竞

例5 51nod 1623 完美消除

- 这种策略难以在数位上进行
- 能否按从左到右的顺序动态维护答案，即在右端添加一个数后，可以在之前的基础上快速计算出新的答案
- 单调栈!
- 以 25342 为例
  - 当加入 3 时，5 已经可以确定要和 3 合并了，答案+1，此时栈为 [2,3]
  - 加入最后一个 2 时，栈为 [2,3,4]，将 3、4 与 2 合并后，发现还有 2，说明新加入的这一个 2 可以与之前的 2 一起消除，不会使答案增加

重庆南开信竞

例5 51nod 1623 完美消除

- 在上述做法中，已确定的位的状态由单调栈确定，可以套到数位 dp 上
- 可以用二进制状压表示单调栈，状态数 $2^9 = 512$
- 本题求的是最小操作数等于 k 的数的个数，还需要保存哪些状态?
  - 已确定的位所需的操作次数
  - 剩余所需的操作次数

重庆南开信竞

例6 HDU 5803 Zhu’s Math Problem

求满足下列条件的 (a, b, c, d) 数量模 10^9+7

- $a+c>b+d$
- $a+d \geq b+c$
- $0 \leq a \leq A$
- $0 \leq b \leq B$
- $0 \leq c \leq C$
- $0 \leq d \leq D$

$1 \leq T \leq 10^3$

$1 \leq A, B, C, D \leq 10^{18}$

重庆南开信竞

例6 HDU 5803 Zhu’s Math Problem

- 数位DP也可以处理多个数之间满足一定条件的计数问题

- 这些约束的形式有：加减法、位运算等
- 它们都与数位紧密相关

- 对于位运算，由于每一位之间独立，可以直接进行数位 dp

- 对于加减法，可以参考在纸上列竖式计算的情形，由于进位、借位的存在，状态的设计和转移的讨论相较于位运算更加复杂

重庆南开信息

例6 HDU 5803 Zhu’s Math Problem

$$
\begin{array}{cccccc}
+ & \ldots & \ldots & a & ? \\
- & \ldots & \ldots & b & ? \\
+ & \ldots & \ldots & c & ? \\
- & \ldots & \ldots & d & ? \\
\hline
 & & carry & dif & ?
\end{array}
$$

重庆南开信竞

- 考虑约束 $a+c>b+d$，已经确定位的差值为 $carry$，dif 为当前考虑位的差值，要满足约束，这一位应该要向前一位提供不少于 $-carry$ 的进位
- $dif \in [-18,18]$，只能提供 $[-1,1]$ 的进位数，因此当 $carry \geq 2$，无论后续怎么取，都一定满足约束；当 $carry \leq -2$，不可能满足约束
- 因此 $carry$ 的状态只有 4 种

例6 HDU 5803 Zhu’s Math Problem

- 经过上述提示，有的同学可能已经脑补出了如下代码

```
for (int a = 0; a <= up1; ++a) {
    for (int b = 0; b <= up2; ++b) {
        for (int c = 0; c <= up3; ++c) {
            for (int d = 0; d <= up4; ++d) {
                int dif1 = carry1 * 10 + a + c - b - d;
                int dif2 = carry2 * 10 + a + d - b - c;
                if (dif1 <= -2 || dif2 <= -2) continue;
                dif1 = min(dif1, 2); dif2 = min(dif2, 2);
                ret += dfs(pos-1, dif1, dif2, limit1&&a==up1, ...);
            }
        }
    }
}
```

重庆南开信竞

- *TLE*

例6 HDU 5803 Zhu’s Math Problem

- 考虑优化
- 四重循环下，一次dfs就有高达 $10^4 = 10000$ 的复杂度
- 将 10 进制改为 2 进制，仍然可以这样 dp，四重循环的复杂度降低到了 $2^4 = 16$
- TLE

重庆南开信竞

例6 HDU 5803 Zhu’s Math Problem

- 注意到：本题有 4 个 $limit$
- 在仅有 1 个 $limit$ 时，我们分析过复杂度（分段计算的段数）
- 但在多个 $limit$ 下，当其中一个数受 $limit$ 限制时，就不能存取 dp 值，这会导致分得的段数呈指数级增长
- 此时，将 $limit$ 限制作为状态进行 dp 就显得十分必要了（即使有效期只有一组数据）

重庆南开信竞

例6 HDU 5803 Zhu’s Math Problem

- 4 个 $limit$ 的组合有共计 16 种状态

- 总状态数 $61 * 4 * 4 * 16 = 15616$，每一个状态需要 $2^4 = 16$ 枚举数位的选择，总共有 1000 组数据，总复杂度约为 249,856,000，较为极限

重庆南开信竞

练习题

热身训练7

vjudge.net/contest/454285

密码

digit

重庆南开信竞