程序设计大作业书面报告

完成人：黄维啸（力 2，2012010270）

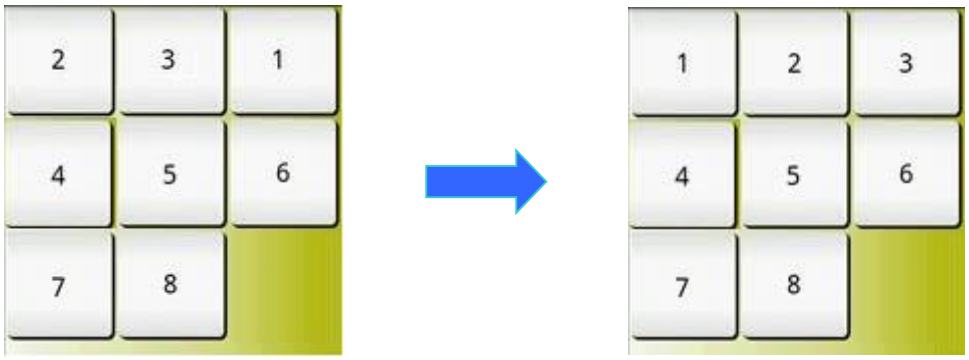
2014 年 1 月 10 日

一、程序计划阶段的初步分析

1、程序名称：

计算机自动重排九宫格——八数组问题的解

2、 程序定义



(a) 初始状态　　(b) 目标状态

如上图所示，在一个 3×3 的区域内放置 8 个滑块。游戏要求从任意初始状态开始，通过依次移动滑块，最终到达目标状态，并且尽量使移动步数最少。

3、 可行性分析

现有的工具有 Fortran 语言和 MATLAB。我们可以考虑用 Fortran 语言来编写相关的程序，运用 Fortran 语言来实现滑块的移动。

运用 Fortran 语言的数组功能可以很好的表示整个九宫格，通过编写程序依次改变数组元素的值，可以把还原九宫格的过程清楚的打印在屏幕上。

因为用 Fortran 编写的程序得到的结果不能非常好的通过图形显示出来，比较单调。于是可以运用 MATLAB 的 GUI 中丰富的图形显示功能编写应用来实现滑块的移动，并把重排就宫格的过程动态的显示出来。

4、 需求分析

本程序需要达到如下功能——

（1）用 Fortran 语言编写移动滑块的程序，具有以下功能：

a） 随机产生初始状态，或用给定的随机数种子重复产生初始状态。

b） 能够判断初始状态是否一定能够移动到目标状态？

c） 若有可行解，给出具体的移动方案，并且统计出移动步数。

（2）用 Matlab 程序编写图形界面，至少具有以下功能：

a） 能够显示初始状态，并报告该是否有到达目标状态的可行解。

b） 能提供一个选项，决定是重新随机生成初始状态还是重复上一次的初始状态。

c） 能够动态地显示滑块每次移动的情况。

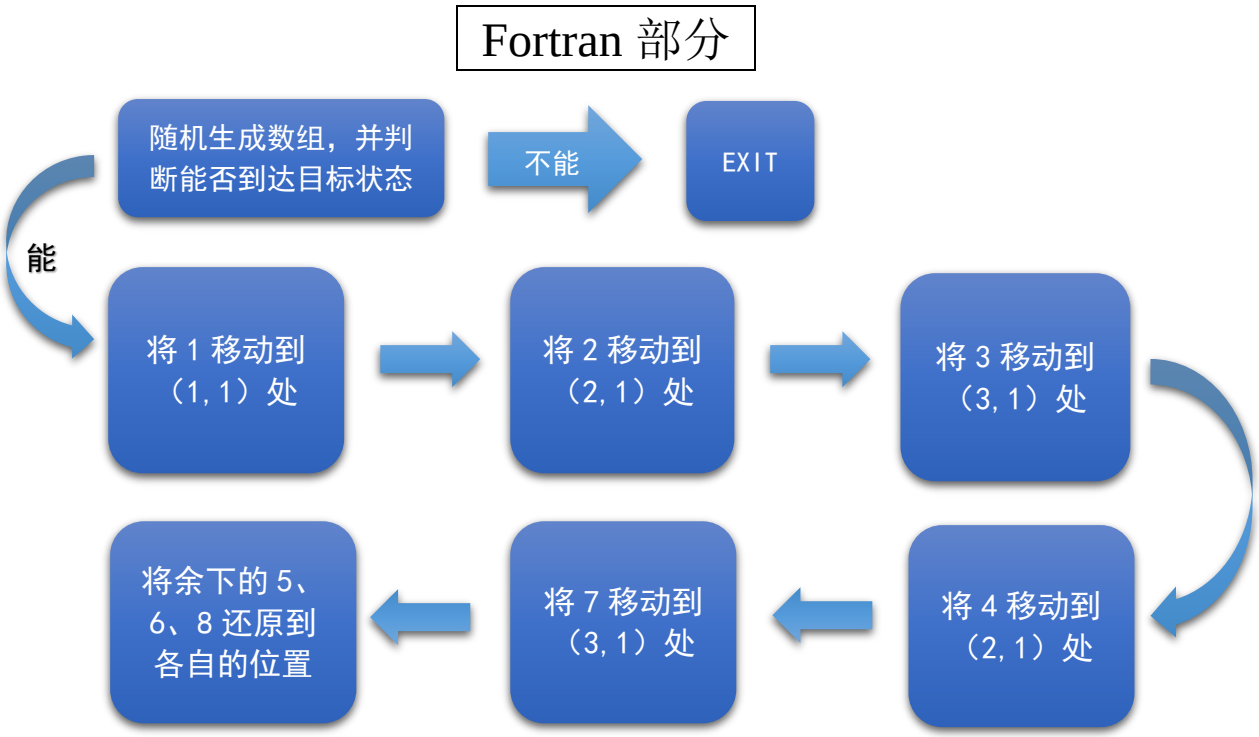
d） 能够显示移动的步数和总共花费的时间。

e） 能够将具体的移动情况保存在磁盘文件中。

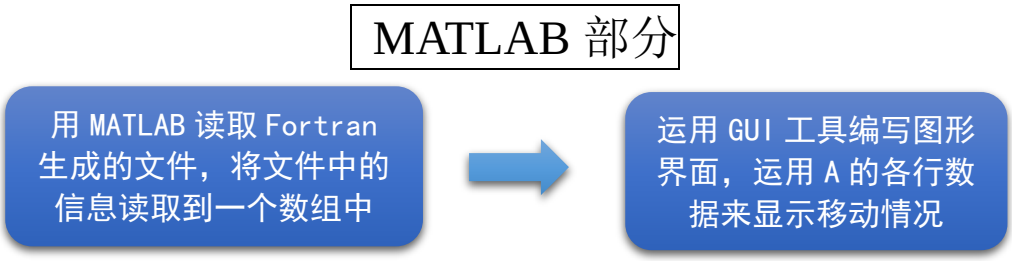
因此需要 Fortran 和 MATLAB 的综合的运用。

二、程序设计思路

1、 概要设计



注：关于将移动过程打印在屏幕上以及记录在文件中的操作，都分配到了每一个操作中了，即每一个框图中都有将移动过程打印在屏幕上以及记录在文件中的操作。



2、 详细设计

• Fortran 部分

为实现移动，将空格用数字 0 来替代，而九宫格的移动方式即为 0 与其周围的元素替换。

重要子程序	功能	算法设计
create_array(c, judge)	随机生成数组 c，并把它打印在屏幕上；接着判断是否能够达到目标状态，能的话就在屏幕上打印“It’s OK!”，并且逻辑变量 judge 的值为 T。否则就在屏幕上打印“Impossible”，judge 的值为 F	1、定义一个一维 9 元素数组 a = (1,2,3,4,5,6,7,8,0)。 2、将 a 的顺序打乱，打乱方法如下：运用循环，将 a 中的元素随机两两对换，循环次数和对换的两个元素的位置用随机种子随机生成，循环多次后 a 就能够被随机打乱了，将打乱后的 a 按照行顺序赋值给一个 3×3 的数组 c，即生成了随机数组。 3、运用逆序数[I]的方法判断 c 能否回到初始状态：从 c 的生成数组 a 着手，不计 a 数组中的数字 0，运用循环计算 a 的逆序数，由于初始状态的 a 为（1,2,3,4,5,6,7,8,0），不计 0 数字的话其逆序数为 0，为偶数，如果计算出打乱后的 a 的逆序数和初始状态相同，即逆序数为偶数，则可以回复到初始状态，否则不能够回复到初始状态。[II]
find_loc(a, x ,m, n)	寻找数字 x 在数组 a 中的位置(m,n)	运用循环跑遍 a 的每一个元素，若有某个元素等于 x，则记录该元素的行列数（m,n）
move_0(a, m, n, k, num)	将数字0移动到3×3阶的数组a的(m,n)处，并且在移动过程中能够自动避开数字 k，即移动过程中不打乱数字k所在的位置。其中num为计步器。每移动一步就打印在屏幕上并记录在文件 record.txt中	1、用子程序 find_loc 记录 0 所在的位置，计算该位置与目标位置（m,n）的行列相对偏差 x,y，若 x 大，则行移动，若 y 大则列移动，移动的结果要使得大的相对偏差绝对值减小 1。就这样可以使得 0 一步一步接近目标位置（m,n） 2、若在 1 中 0 将要移动的时候发现要移动的位置有数字 k，则改为另一个方向移动。即若此时本应该行移动，则改为列移动。

I 逆序数：一个排列中，计算每个数后面比它小的数的个数，将这些个数全部加起来，所得到的的就是排列的逆序数。

II 关于逆序数奇偶性相同就能还原的原因，请参看网页文献 http://www.docin.com/p-33328797.html

move(a, z, m2, n2, num)	将数组 a 中的数字 z 移动到(m2,n2)处，其中 num 为计步器，每移动一步就打印在屏幕上并记录在文件 record.txt 中	1、先找到 z 应该下一步应该移动到的位置，方式和移动 0 的类似，计算行列相对偏差，然后就可以找到下一步应该移动到的位置。 2、调用子程序 move_0 把 0 移动到 z 下一步应该移动到的位置，并且在 0 移动过程中应该避开 z。接着让 0 和 z 交换即可。这样可以让 z 一步步接近目标位置，最终到达目标位置。
move_2(a, num)	将数字 2 移动到目标位置（1,2），其中 num 为计步器。每移动一步就打印在屏幕上并记录在文件 record.txt 中	1、先用子程序 find_loc 找到 2 的位置，然后运用 move 子程序的方法来移动 2 到目标位置（1,2） 2、由于存在特殊情况，故需要特殊情况特别分析。特殊情况为 2 处在中间，即（2,2）的位置的时候，需要特别考虑，具体请看源代码。
move_near(a, m1, n1, m2, n2, num)	将a(m2,n2)上的数字放到a(m1,n1)的相邻右侧或相邻下侧。使用条件如下：(m2-m1==2.and.n2-n1==1) .or. (m2-m1==1.and.n2-n1==2)且a(m1,n1+1)==0，其中num为计步器。每移动一步就打印在屏幕上并记录在文件record.txt中	在(m2-m1==2.and.n2-n1==1)和(m2-m1==1。and.n2-n1==2)两种情况中，可以运用相似的移动方法将（m2,n2）上的数字放到（m1,n1）的相邻右侧或相邻下侧（详细移动方法见子程序源代码）

详细步骤：

（1） 随机生成一个 3×3 的数组 a，并判断其能否到达目标状态：运用子程序 creat_array(a, judge) ，然后若 judge 为 F，则向文件中输出 9 个 1，程序停止；若 judge 为 T，则向文件中输出 9 个 0，程序继续运行。

（2） 将 1 移动到（1,1）处：调用子程序 move(a, 1, 1,1, num)

（3） 将 2 移动到（1,2）处：调用子程序 move_2(a, num)

（4） 将 3 移动到 (1,3)：如果 0 在（1,3）处，就调用子程序 move_0(a, 2,3, -1, num) 将 0 往下移动到（2,3)。若此时 3 已经在目标位置（1,3)，则退出子程序，否则，运用子程序 move(a, 3, 3,3, num) 将 3 移动到(3,3)处，然后运用子程序 move_0(a, 1,3, 2, num) 将 0 移动到（1,3）处，此时满足子程序 move_near 的条件，于是调用子程序 move_near(a, 1,2, 3,3, num) 可以将 3 移动到 2 的右边，即（1,3）处

（5） 将 4 移动到（2,1）处：调用子程序 move(a, 4, 2,1, num)

（6） 将 7 移动到（3,1）处：移动方法与（4）相似，如果 0 在(3,1)处，就调用子程序 move_0(a, 3,2, -1, num) 将 0 往右移动到（3,2)。若此时 7 已经在目标位置（3,1)，则退出子程序，否则，运用子程序 move(a, 7, 3,3, num) 将 7 移动到（3,3）处，然后运用子程序 move_0(a, 3,1, 4, num) 将 0 移动到（3,1）处，此

时满足子程序 move_near 的条件，于是调用子程序 move_near(a, 2,1, 3,3, num)可以将 7 移动到 4 的下面，即（3,1）处。

（7） 将余下的 5、6、8 还原：先运用子程序 move_0(a, 3,3, -1, num)将 0 移动到右下角（3,3）处，然后此时可能会有两种没有还原的情况，如下图所示：

左图可以依次调用子程序 move_0(a, 2,2, 5, num)、move_0(a, 3,3, 6, num)还原

而右图可以依次调用子程序 move_0(a, 2,2, 5, num)、move_0(a, 3,3, 8, num)还原

• MATLAB 部分

（1） 用 MATLAB 来读取 Fortran 生成的文件 record.txt，把文件中数据读取到一个数组 A 中。由于在 Fortran 中已经编写程序把数组的第二行定为判断行，即若九宫格可以还原，则第二行全部为 0，否则全部为 1。于是在 MATLAB 中可以凭借 A 的第二行来确定程序是否要继续下去。若 A 第二行全为 0，则程序继续。

（2） 运用 GUI 控件工具编写图形界面，画出 9 个 handles 来表示九个格子，同时再加上标题，时间、按钮等工具，然后在 MATLAB 中编写循环，使得 9 个格子中的数字分别对应了数组 A 中每一行的元素，运用 pause 语句每 0.5 秒换一行，这样就可以把 Fortran 中生成的移动步骤用 9 个格子每 0.5 秒的变化来表示。

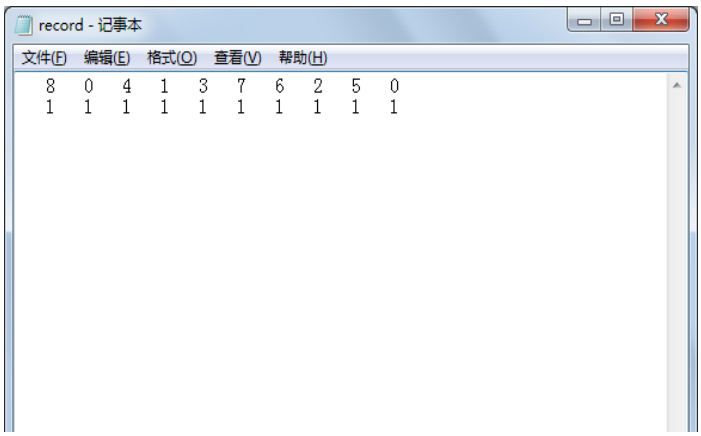
三、程序测试

1、 测试方案设计

先运行 Fortran 的 exe 文件，观察文件中的输出结果和 MATLAB 的 exe 文件的结果，若可行，再点击 MATLAB 的 exe 文件中的 Start 观察移动情况，达到目标情况后，分别点击 Return 和 Generate 来看有什么变化。

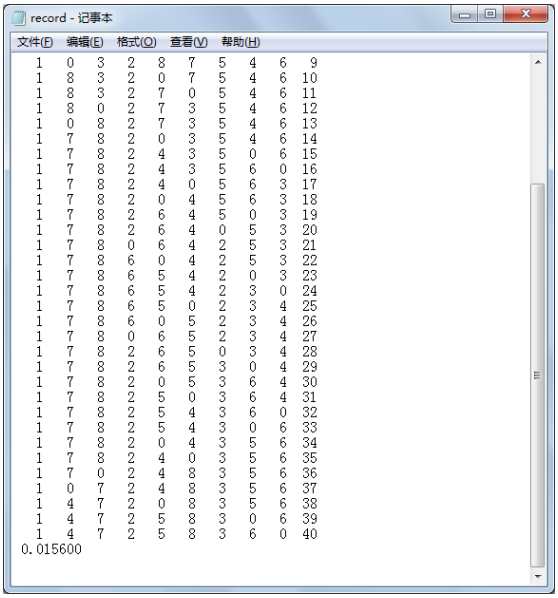
2、 测试结果

先运行 Fortran 中的 exe 文件，得到文件 record.txt，打开可以看到如下左图：



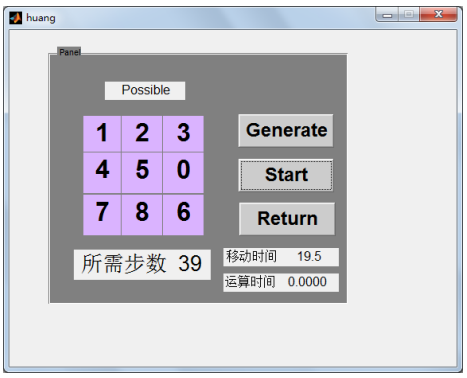
此时显示的是不能达到目标位置，第一行是按列来储存数组 a 的初始值。而上图右图就是此时 MATLAB 中 exe 文件所显示的内容。

再次运行 Fortran 的 exe 文件，更新文件 record.txt，打开可以看到如下左图：



此时每一行前九列显示了数组 a 的还原移动情况，第十列是移动步数，而最后的 0.015600 是 Fortran 程序的运行时间。

上图右图是此时对应的 MATLAB 中 exe 文件显示的内容，此时问题可解。点击 Start 开始动态还原，如下：



右下角就是还原后的情况，上面显示了所需的步数、移动时间和 Fortran 程序的运算时间。如果此时点击 Return，则有回到了还原前的初始状态，如下左图。而若点击 Generate 键，则就重新生成一个九宫格并判断其能否成功还原，如下右图：

四、心得体会

这次大作业我觉得自己做得特别带劲。从最先的不知从何下手，到最后自己独立编写然后把 3 阶的移动程序推广到 n 阶，再到最后优化自己的 n 阶的 Fortran 程序，整个过程自己收获了很多，而且发现自己对于编程有了更加浓厚的兴趣。编写完九宫格后，我发现自己的算法可以推广到 n 阶的情况，于是就花好几天的时间，终于编写一个“n 宫格”的 Fortran 程序。运行时，使用者可以根据输入的 n 的个数来随机生成一个 n 阶数组，判断其能否达到目标状态，如果可以，就利用我的算法来实现 n 阶数组的还原。

自己写的 3 阶的代码总来长度是 300 多行，而 n 阶的代码主程序+模块总共也只是 500 多行，听同班同学说，他们所编写的程序都是六七百、七八百行，有的甚至一千多行，自己心中也是非常高兴的。

在后来自己由对 n 阶的程序进行了优化，发布了 2.0 的版本，并添加了一些功能，其实我自己也想用寒假的时间努力学习 MATLAB，想运用 MATLAB 把我的 n 阶的数组的还原步骤动态显示出来。可以说，这次大作业让我对于编程这方面有了更大的兴趣，同时

也坚定了我假期想要努力学习C语言等更多程序语言的决心。

总的来说，这次大作业对我的影响挺大的，希望以后也能够多多这样练习一下，提高自己的码代码的能力。