Solving Sudoku Cons – The Sudoku assistant:

The Sudoku assistant was originally developed to solve the Literal Hyper Sudoku (Cm JA 2011 cover) by XAMAN EK, after a couple of weeks of trying to do it manually. Once that problem was solved, it was fairly easy to adapt the solver to solve the standard 9x9 sudokus found in Cm. Previously, I was using the ISANAKI solver, and a nice Excel solver for 9x9 sudokus, by Andy Pope. While I was at it, I took a bit of time to eliminate the painful task of converting letters to numbers, manually loading numbers into the solver, and after the solver does its job, converting the numbers to letters. With this solver, you can use the digital cons as-is, and see the results as text.

The vast majority of Sudoku problems published in newspapers, even those labeled "hard" or "challenger", require only basic to intermediate solving strategies. The most useful tool you can have is a quick way to calculate the candidates for any given cell. An easy way to calculate candidates for any given cell works as follows: remove all characters found in the same row, column, and 3x3 (or 4x4) square from the list of all characters (A-Z) used by the con. In the case of the hyper-sudoku on the JA 2011 CM cover, all the grey cells have additional constraint as follows: the characters in the cells on the main diagonals must also be removed from the candidates on that diagonal, and all the characters in one of the grey squares must be removed from the candidates in that square.

While this can be done with standard worksheet functions, the following user-defined functions will simplify the job of calculating candidates
1. =mmconc(A16:I16,TRUE) will return a string that contains all the characters in the range (row 1), with minus signs for empty cells. This may help you guess at the key when the program gives you a partial solution.

2. =nodupchar(MMconc(A16:I24, False)) will return all the characters used by the con, without minus signs or duplicate letters. This particular value has been named chSET. CAUTION: It is possible to solve sudokus where one character is missing. This program does not handle this case.

The candidates for cell 1 in the standard 9x9 sudoku can then be calculated as =IF(A16="",map(chSet,P28&sq91&srow91&scol91,""),""), where P28 is a list of characters that cannot be candidates (as determined by intermediate solution strategies), sq91 is the list of characters in the first 3x3 square, srow91 are those found in row 1, and scol91 are those which are found in column1. Here, sq91 is the name I have assigned to cell L16, srow91 to cell M16, and so on. For the center cell of the candidate table, the formula becomes =IF(E20="",map(chSet,T32&sq95&srow95&scol95,""),""). This makes creating the candidates table (P16:X24) much easier. The table is the most critical part of this solver. Errors will result in endless grief.

Once you have the candidate table set-up, solving becomes almost easy: some candidates will be one letter long. This is the "one-choice" situation. Copy the "one-choices" to the input block (A16:P24). Quite often this will produce additional "one-choices". Next, we do “scanning”: if a letter occurs once and only once in any row, column or 3x3 square, then that letter must be the correct candidate for the cell in which it is found, no matter how many other letters are shown as possible candidates. All this work is done for you automatically.

Intermediate and advanced strategies for solving Sudoku problems are described in detail in the MENSA guide to solving Sudoku. Also, you can find a Sudoku solver at http://www.pennydellsudokusolver.com that allow you to interactively solve numeric sudokus found in Dell Sudoku magazines at most newsstands. The beauty of this solver is that it can provide assistance, with explanations for a variety of advanced techniques. As time permits, I intend to include more of these techniques in this program as needed to solve the tougher problems. Alternatively, I may implement some variation of BION’s brute force algorithm to complete the solutions, see BION’s website.

Using the Sudoku assistant is easy: copy the digital con and paste it into cells A1 and A2 of the Sudoku9x9 worksheet. This updates the table in cells A5:I13.Next, press the Initialize Sudoku button. This updates the input block (A16:I24), performs some diagnostics, and clears the “Cannot be” table. Finally, press the “Solve Sudoku” button, and see the solution appear, or at least, part of the solution. Also, on the SolutionPath worksheet, you will find a listing of the steps used by the computer to reach the given solution. This was, at first, a tool to help diagnose bugs. Now, I find it a handy method of determining what I missed while attempting to solve sudokus manually.

At this point, look at the tables in cells L16:N37. This will show the possible solutions, or part of them. If you are told that the solution appears backward, look at L29:N37, otherwise, look at L16:N24. If you cannot yet see the right one, look at the candidates table, and pick one of the shorter ones. Pick one of the possible choices, and paste into the appropriate cell of the input block. Be sure to use upper case letters. Next try your luck by pressing the “Solve Sudoku” button again. Keep track of your choices, in case you need to try again. You will have a good solution when every cell in the input block contains a single character.

This program can also solve numeric 9x9 sudokus. However, you will have enter the number into the input block manually. Be sure you clear the “Cannot Be” table before pressing the “Solve Sudoku” button.

The Sudoku assistant has been posted on the ACA website, under Resources /Computer Column Programs. On the HYPER sheet, you will find the tables for the Literal Hyper Sudoku (Cm JA 2011 cover) by XAMAN EK. The formulas on this sheet mostly use standard worksheet functions. Compare them with the formulas on the Sudoku9x9 sheet, and you will understand the advantage of using user – defined functions. Press the Solve Hyper Sudoku button, and see the solution appear in square 10.

The VBA (Visual Basic for Applications) code for the user-defined functions and the Solver routines can be viewed by pressing ALT-F11.

On the CMSudoku sheet, you will find, a compilation of all the Sudoku cons published in Cm to date. This helped a lot while I was figuring out problems and fixing bugs. Thanks to PARROT for compiling the list for me. Over 95% of these cons can be solved using only basic strategies