Candidate Separation Analysis:

by G.A. Aldrich

Version 1.0, May 23, 2015

Introduction

Recently, I was glancing over a sudoku which had gone through the pre-annotational addition of solved squares, then the process of annotation itself, then box-line eliminations, and group separation. I was planning on the usual sequence of possible n-wing reductions followed by further group separation, followed by a solution (if necessary) by subpattern analysis.

As I looked over the first row, I wondered idly what the solved values in the unsolved squares might be. A possible sequence of candidates crossed my mind, and I considered for a mad minute of inserting that particular sequence and trying out an update with them, but I desisted, realizing the unlikeliness of it solving the sudoku. It got me to wondering, however, how many different sets of choices existed for that row. It should be easy to find out, simply by calculating all of them and listing them in an orderly and exhaustive fashion.

That brought up the idea that if I tested each set of choices by simply plugging them into that first row, and then were able to update the rest of the sudoku with a temporary update (as in subpattern analysis), all of the wrong choices might lead to contradictions, and the right choice to a solution. just as in subpattern analysis.

I will limit myself to speaking about rows, but what I say applies equally to columns and boxes. This will save on verbiage. In addition, I will use the term ***separation*** to refer to each set of choices.

The main consideration in choosing the right row is the avoidance of an overly large number of possible separations to test out, until the winning separation is found.. This requires diligence in finding every possible separation, lest the right one not be discovered, despite the fact that updating each separation could be time-consuming.

It is probably easiest just to look at the solutions to the sudokus solved by this method, rather than be subjected to a lengthy abstract explanation. For naming conventions, etc., check in the appendix..

The majority of the sudokus which appear in the examples of this text were thoroughly tested for the existence of n-wings (my generalized terminology for x-wings, swordfish, jellyfish, and squirmbags), as well as for polarity situations. All of them have been through countless group separations and box-line reductions, in a thorough process of initial solving prior to resorting to the method presented in this text.

Gary Aldrich

Table of Contents

Sudoku #1

The sudoku below has undergone all the steps of solving - trapping[1], annotation, box-line reductions[2], and group separations[3]. Row 1 is chosen as the basis for candidate separation analysis.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►► row1	23	35	7	25	9	4	6	8	1
row2	48	45	6	7	1	58	3	2	9
row3	28	9	1	28	3	6	4	5	7
row4	3479	1	8	359	6	2	57	37	45
row5	3479	34	5	139	8	17	2	367	46
row6	37	6	2	35	4	57	1	9	8
row7	6	7	3	18	5	18	9	4	2
row8	1	8	9	4	2	3	57	67	56
row9	5	2	4	6	7	9	8	1	3

Separation Analysis Table:

col 1	col 2	col 4		
23	35	25		
2	3*	5*	sep1	(2 forces 5 forces 3)
3	5*	2*	sep2	(3 forces 5 forces 2)

sep = separation
* = forced choice, underlined = free choice
(It took me less than 10 seconds to calculate the two separations)

Sudoku #1, continued.

We shall first test separation 1:

2 3 5

		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►►	row1	23 **2**	35 **3**	**7**	25 **5**	**9**	**4**	**6**	**8**	**1**
	row2	48 **4**	45 **5**	**6**	**7**	**1**	58 **8**	**3**	**2**	**9**
	row3	28 **8**	**9**	**1**	28 **2**	**3**	**6**	**4**	**5**	**7**
	row4	3479 **3**	**1**	**8**	359 **9**	**6**	**2**	57 **5**	37 **7**	45 **4**
	row5	3479 **9**	34 **4**	**5**	139 **1**	**8**	17 **7**	**2**	367 **3**	46 **6**
	row6	37 **7**	**6**	**2**	35 **3**	**4**	57 **5**	**1**	**9**	**8**
	row7	**6**	**7**	**3**	18 **8**	**5**	18 **1**	**9**	**4**	**2**
	row8	**1**	**8**	**9**	**4**	**2**	**3**	57 **7**	67 **6**	56 **5**
	row9	**5**	**2**	**4**	**6**	**7**	**9**	**8**	**1**	**3**

It took me less than 2 minutes to update this sudoku with separation 1, using a temporary update. All rows, columns, and boxes are 9-perfect, so it is the solution. Therefore separation 2 does not need to be tested.

Final Report:

col 1	col 2	col 4	sep#	result	time
23	35	25		table	20 sec
2	3*	5*	sep1	solves	2 min
3	5*	2*	sep2	-------	
				total time	2 min

* = forced choice, underlined = free choice

Sudoku #2

The sudoku below has undergone all the steps of solving - trapping, annotation, row-box reductions, column-box reductions, and group separations. Row 4 is chosen as the basis for the separation analysis.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	9	5	16	136	67	8	467	1347	2
row2	8	4	7	1356	2	135	56	135	9
row3	12	26	3	4	9	57	57	8	16
►► row4	6	29	58	58	1	4	3	29	7
row5	12	3	58	7	56	9	2468	24	16
row6	7	19	4	68	3	2	68	19	5
row7	4	8	129	159	57	6	2579	257	3
row8	5	67	269	39	4	37	1	27	8
row9	3	17	19	2	8	157	579	6	4

Separation Analysis Table:

	col 2	col 3	col 4	col 8	sep#
row 4	29	58	58	29	
	2	5	8*	9*	sep1
	2	8	5*	9*	sep2
	9	5	8*	5*	sep3
	9	8	5*	2*	sep4

Sudoku #2, continued.

I have discovered that a single pair of pairs may be left as it is, and not broken down into its component digits. This omission has, at least in my experience, no ill effect in the solving power of the resultant list of separations. If we follow this advice in the present problem, the separation table is simplified:

Original Separation Table:

	col 2	col 3	col 4	col 8	sep#
row 4	29	58	58	29	
	2	5	8*	9*	sep1
	2	8	5*	9*	sep2
	9	5	8*	5*	sep3
	9	8	5*	2*	sep4

Revised Separation Table:

In the table below, sep1 and sep2 are called ***combined separations***. The new sep1 represents both the old sep1 and sep2, and the new sep2 represents both the old sep3 and sep4 The purpose of using them is to reduce the number of separations to be tested. We shall go into more detail about combined separations later. For the moment, we'll simply test them in the same way as we do ordinary separations.

	col 2	col 3	col 4	col 8	sep#
row 4	29	58	58	29	
	2	58	58	9*	sep1
	9	58	58	2*	sep3

We'll make one further simplification to the revised table:

Simplified Separation Table:

	col 2	col 8	sep#
row 4	29	29	
	2	9*	sep1
	9	2*	sep3

In the simplified table, we're simply omitting the pair of 58's, since inserting them into the sudoku accomplishes nothing.

We'll be using the simplified separations for testing this particular sudoku..

Sudoku #2, continued.

We shall first test separation 1.

		2						**9**	
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	9	5	16 1	136 3	67 6	8	467 4	1347 7	2
	row2	8	4	7	1356 5	2	135 1	56 6	135 3	9
	row3	12 2	26 6	3	4	9	57 7	57 5	8	16 1
►►	row4	6	29 2	58 5	58 8	1	4	3	29 9	7
	row5	12 1	3	58 8	7	56 5	9	2468 2	24 4	16 6
	row6	7	19 9	4	68 6	3	2	68 8	19 1	5
	row7	4	8	129 2	159 1	57 7	6	2579 9	257 5	3
	row8	5	67 7	269 6	39 9	4	37 3	1	27 2	8
	row9	3	17 1	19 9	2	8	157 5	579 7	6	4

All rows, columns, and boxes are 9-perfect, so this is the solution.

Final Report:

	col 2	col 8	sep#	result	time
row 4	29	29		table	negligible
	2	9*	sep1	solves	4 min
	9	2*	sep2	-------	
				verify	2 min
				total	6 min

Sudoku #3

The sudoku below has undergone all the steps of solving - trapping, annotation, row-box reductions, column-box reductions, and group separations. Row 8 is chosen as the basis for separation analysis.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	9	78	678	146	2	147	3	18	5
row2	57	4	1	8	57	3	6	9	2
row3	5678	3	2	9	567	157	17	4	178
row4	478	1	3	245	9	245	2457	6	478
row5	2	5	48	7	3	6	14	18	9
row6	467	9	467	1245	8	1245	12457	3	147
row7	148	2	489	56	456	458	149	7	3
►► row8	147	6	5	3	47	9	8	2	14
row9	3	78	4789	24	1	2478	49	5	6

Separation Table.

	col 1	col 5	col 9		result	time
row 8	147	47	14		table	20 sec
	1	7*	4*	sep1		
	4	7*	1*	sep2		
	7	4*	1*	sep3		

* = forced choice, underlined = free choice

time to calculate table = approximately 20 seconds.

Sudoku #3, continued.

We first test separation 1:

		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
		1				**7**				**4**
	row1	9	78 8	678 6	146 4	2	147 7	3	18 1	5
	row2	57 7	4	1	8	57 5	3	6	9	2
	row3	5678	3	2	9	567 6	157 1	17 7	4	178 8
	row4	478 4	1	3	245 5	9	245 2	2457 5	6	478 7
	row5	2	5	48 8	7	3	6	14 4	18 8	9
	row6	467 6	9	467 7	1245	8	1245	12457 25	3	147 1
	row7	148 8	2	489	56 6	456 4	458 5	149 1	7	3
►►	row8	147 1	6	5	3	47 7	9	8	2	14 4
	row9	3	78 7	4789 4	24 2	1	2478 8	49	5	6

Separation 1 fails in 3 minutes with duplicate 5's in row 4.

	col 1	col 5	col 9		result	time
row 8	147	47	14			
	1	7*	4*	sep1	fails	3 min
	4	7*	1*	sep2		
	7	4*	1*	sep3		

Sudoku #3, continued.

We next test separation 2.

		4				**7**			**1**
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	9	78 8	678 6	146 4	2	147 7	3	18 1	5
	row2	57 7	4	1	8	57 5	3	6	9	2
	row3	5678 5	3	2	9	567 6	157 1	17 7	4	178 8
	row4	478 8	1	3	245 5	9	245 4	2457 2	6	478 7
	row5	2	5	48 4	7	3	6	14 1	18 8	9
	row6	467 6	9	467 7	1245 1	8	1245 2	12457 5	3	147 4
	row7	148 1	2	489 8	56 6	456 4	458 5	149 9	7	3
►►	row8	147 4	6	5	3	47 7	9	8	2	14 1
	row9	3	78 7	4789 9	24 2	1	2478 8	49 4	5	6

Separation 2 solves the sudoku in 5 min. Everything is 9-perfect.

Final Results:

	col 1	col 5	col 9		result	time
row 8	147	47	14		table	20 sec
	1	7*	4*	sep1	fails	3 min
	4	7*	1*	sep2	solves	5 min
	7	4*	1*	sep3	-------	
					verify	2 min
					total	10 min

Sudoku #4

The sudoku below has undergone all the steps of solving - trapping, annotation, row-box reductions, column-box reductions, and group separations. Row 1 is chosen for separation analysis with squares (15), (16), (18) & (19). Note that we are only using a partial row, omitting the pair of 17's.

#4		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►►	row1	17	17	2	8	4569	49	3	69	4569
	row2	46	9	3	1	2456	7	245	8	2456
	row3	5	46	8	46	3	29	1	29	7
	row4	2467	3	46	9	47	5	247	1	8
	row5	12479	1457	1459	47	8	3	2479	2679	2469
	row6	479	8	49	2	1	6	479	5	3
	row7	3	456	4569	4567	24679	249	8	279	1
	row8	169	2	169	3	679	8	579	4	59
	row9	8	45	7	45	29	1	6	3	29

Separation Table:

	col 5	col 6	col 8	col 9		result	time
row 1	4569	49	69	4569		table	5 min
	4	9*	6*	5*	sep1		
	5	4*	69	69*	sep2		
	69	4*	69*	5*	sep3		

Note the use of combined separations. We'll be explaining more about them after the completion of solving Sudoku #4.

time to calculate logic table = 5 min

Sudoku #4, continued.

We first test separation 1.

					4	**9**		**6**	**5**
#4		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►►	row1	17	17	2	8	4589 4	49 9	3	69 6	4569 5
	row2	46 6	9	3	1	2456 5	7	245 24	8	2456 24
	row3	5	46 4	8	46 6	3	29 2	1	29 9	7
	row4	2467 24	3	46 6	9	47 7	5	247 24	1	8
	row5	12479	1457 17	1459	47	8	3	2479 79	2679	2469
	row6	479	8	49	2	1	6	479 79	5	3
	row7	3	456 6	4569 4	4567	24679	249	8	279	1
	row8	169 19	2	169 19	3	679 67	8	579 5	4	59 9
	row9	8	45 5	7	45	29	1	6	3	29

After only 2 minutes, a contradiction occurs in row 8, showing that separation 1 fails.

	col 5	col 6	col 8	col 9		result	time
row 1	4569	49	69	4569		table	5 min
	4	9*	6*	5*	sep1	fails	2 min
	5	4*	69	69*	sep2		
	69	4*	69*	5*	sep3		

Sudoku #4, continued.

We next test separation 2.

					5	**4**		**69**	**69**
#4		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►►	row1	17	17	2	8	4569 5	49 4	3	69 6	4569 9
	row2	46 6	9	3	1	2456 2	7	245 45	8	2456 45
	row3	5	46 4	8	46 6	3	29 9	1	29 2	7
	row4	2467 247	3	46 6	9	47 4	5	247 2	1	8
	row5	12479 1247	1457 17	1459	47 7	8	3	2479 49	2679 7	2469 6
	row6	479 47	8	49 9	2	1	6	479 49	5	3
	row7	3	456 6	4569 4	4567 5	24679 7	249 2	8	279 9	1
	row8	169 9	2	169 1	3	679 6	8	579 7	4	59 5
	row9	8	45 5	7	45 4	29 9	1	6	3	29 2

Separation 2 takes up 4 minutes before it hits a snag with two 7's in row 5.

	col 5	col 6	col 8	col 9		result	time
row 1	4569	49	69	4569		table	5 min
	4	9*	6*	5*	sep1	fails	2 min
	5	4*	69	69*	sep2	fails	4 min
	5	9*	6*	4*	sep3		
	69	4*	69*	5*	sep3		

Sudoku #4, continued.

We next test separation 3

						5	**9**		**6**	**4**
#4		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►►	row1	17	17	2	8	4569	49	3	69	4569
	row2	46	9	3	1	2456	7	245	8	2456
	row3	5	46	8	46	3	29	1	29	7
	row4	2467	3	46	9	47	5	247	1	8
	row5	12479	1457	1459	47	8	3	2479	2679	2469
	row6	479	8	49	2	1	6	479	5	3
	row7	3	456	4569	4567	24679	249	8	279	1
	row8	169	2	169	3	679	8	579	4	59
	row9	8	45	7	45	29	1	6	3	29

.

After 6 minutes, separation 3 solves the grid, and verification shows the grid to be 9-perfect, so the grid may rightfully be called a sudoku.

	col 5	col 6	col 8	col 9		result	time
row 1	4569	49	69	4569		table	5 min
	4	9*	6*	5*	sep1	fails	2 min
	5	4*	69	69*	sep2	fails	4 min
	69	4*	69*	5*	sep3	solves	7 min
						verify	2 min
						total	20 min

Some remarks on combined separations:

When any kind of separation ***fails***, it means that a contradiction has occurred. A ***contradiction*** consists of any situation in which 9-perfection has been violated. There are a number of ways in which this violation could be expressed – a fatal four is one, or two squares in the same row, column or box might have the same value, or three squares in the same row, column or box might have only two candidates or values in common. When a contradiction occurs, the separation cannot solve the sudoku. It has failed. If it has failed, it is always because a contradiction has occurred.

When a simple separation ***bogs down***, it means that the separation cannot be taken any further in testing the sudoku, because it cannot simplify further squares. It means that the row used for the separation does not have sufficient information to complete the testing.

The rest of this page refers only to ***combined separation***s.

When a combined separation bogs down, it should be broken down into its two members, and each of them tested separately. If both of them bog down, then another row, probably with more candidate squares, should be used as a basis for a more extensive separation.

A ***combined separation*** always represents two ordinary separations, which we'll call ***members***. The following rules represent my experience.

Rule 1: If a combined separation succeeds, then
one of its members will solve the sudoku when tested by itself
and the other member will fail if tested all by itself.

Rule 2: If a combined separation fails, then
both members will fail if individually tested.

Rule 3: If a combined separation bogs down, and
if one of its members fails, then
if the other member does not also bog down,
it (the other member) will solve the sudoku
but if it (the other member) also bogs down,
it is correct as far as it goes
but the combination separation is not adequate enough,
and a more extensive separation should be attempted.

Note 1: When a combined separation solves a sudoku, you can always tell which of its two members does the solving. It is the one which agrees with the solution in the two squares which originally contained the candidate pair.

Note 2: You cannot omit the pair, creating the Simplified Separation Table, as was done in Sudoku #2, unless the pair exists, as a pair, in the original row. To do otherwise creates disastrous errors in all the subsequent logic, because the pair exerts selective power, disallowing all other candidate values that are in the original squares.

Sudoku #5

The sudoku below has undergone all the steps of solving - trapping, annotation, row-box reductions, column-box reductions, and group separations. Row 6 is chosen for separation analysis.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	136	1359	1356	4	2	8	125	7	135
row2	4	179	126	3	679	5	12	69	8
row3	2367	3579	8	69	679	1	2369	69	4
row4	23	4	9	1	8	23	7	5	6
row5	123	8	123	7	5	6	39	4	239
►► row6	5	6	7	29	39	4	8	269 1	239
row7	9	1357	4	8	137	237	6	23	157
row8	8	1357	1356	26	1367	9	4	23	157
row9	1367	2	136	5	4	37	19	8	179

	col 4	col 5	col 9		result	time
row 6	29	39	239		table	2 min
	2	39	39	sep1		
	9	3*	2*	sep2		

Sudoku #5 continued

We first test separation 1:

		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
					2	**39**				**39**
	row1	136	1359 9	1356 36	4	2	8	125 5	7	135 1
	row2	4	179 7	126 16	3	679 6	5	12 2	69 9	8
	row3	2367 2	3579 5	8	69 9	679 7	1	2369 3	69 6	4
	row4	23 2	4	9	1	8	23	7	5	6
	row5	123 13	8	123 13	7	5	6	39 9	4	239 2
►►	row6	5	6	7	29 2	39 9	4	8	269 1	239 3
	row7	9	1357 13	4	8	137 1	237 2	6	23 3	157 57
	row8	8	1357 13	1356 5	26 6	1367 3	9	4	23 2	157 57
	row9	1367 7	2	136 6	5	4	37 7	19 1	8	179 9

Separation 1 fails in 4 min with a pair of 2's in column 1.

	col 4	col 5	col 9			
row 6	29	39	239		table	2 min
	2	39	39	sep1	fails	4 min
	9	3*	2*	sep2		

Sudoku #5 continued

We next test separation 2:

		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
					9	**3**				**2**
	row1	136 6	1359 9	1356 3	 4	 2	 8	125 5	 7	135 1
	row2	 4	179 7	126 1	 3	679 9	 5	12 2	69 6	 8
	row3	2367 2	3579 5	 8	69 6	679 7	 1	2369 3	69 9	 4
	row4	23 3	 4	 9	 1	 8	23 2	 7	 5	 6
	row5	123 1	 8	123 2	 7	 5	 6	39 9	 4	239 3
►►	row6	 5	 6	 7	29 9	39 3	 4	 8	269 1	239 2
	row7	 9	1357 3	 4	 8	137 1	237 7	 6	23 2	157 5
	row8	 8	1357 1	1356 5	26 2	1367 6	 9	 4	23 3	157 7
	row9	1367 7	 2	136 6	 5	 4	37 3	19 1	 8	179 9

Separation 2 solves the sudoku in 9 minutes. All is 9-perfect.

Final Results:

	col 4	col 5	col 9			
row 6	29	39	239		table	2 min
	2	39	39	sep1	fails	4 min
	9	3*	2*	sep2	solves	9 min
					verify	2 min
					total	17 min

Sudoku #6

The sudoku below has undergone all the steps of solving - trapping, annotation, row-box reductions, column-box reductions, and group separations. Row 1 is chosen as the basis for the separation analysis.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►► row1	4	256	257	8	9	267	1	26	3
row2	289	3	278	1467	1267	1247	5	2689	26789
row3	2589	2569	1	3567	2567	2367	6789	2689	4
row4	12359	1259	6	159	4	19	389	7	2589
row5	359	7	4	2	56	8	369	1	569
row6	1259	8	25	15679	3	1679	4	269	2569
row7	7	145	358	13469	16	13469	2	34	689
row8	18	14	9	13467	1267	123467	678	5	678
row9	6	24	23	79	8	5	79	34	1

Separation Calculation Table:

	col 2	col 3	col 6	col 8		
row 1	256	257	267	26	table	2 min
	2	5*	7*	6*	sep1	
gen1	5	27	267	26		
	5	2	7*	6*	sep2	
	5	7	26	26	sep3	
	6	5*	7*	2*	sep4	

Sudoku #6, continued.

We will first test separation 1.

		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
			2	**5**			**7**		**6**	
►►	row1	4	256 2	257 5	8	9	267 7	1	26 6	3
	row2	289 8	3	278 7	1467 146	1267 16	1247 14	5	2689 2	26789 9
	row3	2589 9	2569 6	1	3567 35	2567 25	2367 23	6789 7	2689 8	4
	row4	12359 1235	1259 9	6	159	4	19	389 38	7	2589 258
	row5	359 35	7	4	2	56	8	369 36	1	569 56
	row6	1259 125	8	25	15679	3	1679	4	269 9	2569 256
	row7	7	145 5	358 3	13469 169	16	13469 169	2	34 4	689 8
	row8	18 8	14 1	9	13467 34	1267 2	123467 34	678 6	5	678 7
	row9	6	24 4	23 2	79 7	8	5	79 9	34 3	1

Separation 1 fails in 5 minutes with duplicate 8's in column 1.

	col 2	col 3	col 6	col 8		result	time
row 1	256	257	267	26		table	2 min
	2	5*	7*	6*	sep1	fails	5 min
gen1	5	27	267	26			
	5	2	7*	6*	sep2		
	5	7	26	26	sep3		
	6	5*	7*	2*	sep4		

Sudoku #6, continued.

We next test separation 2.

		5	**2**			**7**		**6**	
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►►	row1	4	256 5	257 2	8	9	267 7	1	26 6	3
	row2	289 89	3	278	1467	1267	1247	5	2689	26789
	row3	2589 89	2569	1	3567	2567	2367	6789	2689	4
	row4	12359	1259	6	159	4	19	389	7	2589
	row5	359	7	4	2	56	8	369	1	569
	row6	1259	8	25	15679	3	1679	4	269	2569
	row7	7	145 14	358	13469	16	13469	2	34	689
	row8	18 1	14	9	13467	1267	123467	678	5	678
	row9	6	24	23	79	8	5	79	34	1

After only 1 minute, 3 squares in box G have only 2 candidates, showing separation 2 to be invalid..

	col 2	col 3	col 6	col 8		result	time
row 1	256	257	267	26		table	2 min
	2	5*	7*	6*	sep1	fails	5 min
gen1	5	27	267	26			
	5	2	7*	6*	sep2	fails	1 min
	5	7	26	26	sep3		
	6	5*	7*	2*	sep4		

Sudoku #6, continued.

We next test separation 3.

		5	**7**			**26**		**26**	
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►►	row1	4	256 5	257 7	8	9	267 6	1	26 2	3
	row2	289 2	3	278 8	1467 1	1267 7	1247 4	5	2689 9	26789 6
	row3	2589 9	2569 6	1	3567 3	2567 5	2367 2	6789 7	2689 8	4
	row4	12359 3	1259 9	6	159 5	4	19 1	389 8	7	2589 2
	row5	359 5	7	4	2	56 6	8	369 3	1	569 9
	row6	1259 1	8	25 2	15679 9	3	1679 7	4	269 6	2569 5
	row7	7	145 4	358 5	13469 6	16 1	13469 9	2	34 3	689 8
	row8	18 8	14 1	9	13467 4	1267 2	123467 3	678 6	5	678 7
	row9	6	24 2	23 3	79 7	8	5	79 9	34 4	1

Separation 3 solved the sudoku in 10 minutes, with everything 9-perfect.

	col 2	col 3	col 6	col 8		result	time
row 1	256	257	267	26		table	2 min
	2	5*	7*	6*	sep1	fails	5 min
gen1	5	27	267	26			
	5	2	7*	6*	sep2	fails	1 min
	5	7	26	26	sep3	solves	10 min
	6	5*	7*	2*	sep4	no test	------
						verify	2 min
						total	20 min

Sudoku #7

The sudoku below has undergone all the steps of solving - trapping, annotation, row-box reductions, column-box reductions, and group separations. Row 2 is chosen as the basis for the separation analysis.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	245	1	9	278	3	28	6	25	47
►► row2	256	7	3	26	4	1	9	8	25
row3	8	24	26	2679	5	29	47	3	1
row4	259	3	258	4	18	7	125	12569	256
row5	1	6	4	25	9	25	8	7	3
row6	2579	258	2578	3	18	6	1245	1259	245
row7	3	2458	2568	158	7	458	125	1256	9
row8	257	9	2578	158	6	3	1257	4	2578
row9	4567	458	1	589	2	4589	3	56	5678

	col 1	col 4	col 9	sep#	result	time
row2	256	26	25		table	1 min
	25	6*	25	sep1		
	6	2*	5*	sep2		

Sudoku #7, continued.

We'll first test separation 1:

		25		**6**						**25**
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	245 25			278		28		25 2	47 4
►►	row2	256 25			26 6					25 5
	row3		24 4	26 6	2679 29		29	47 7		
	row4	259 9		258 28		18		125 5	12569 19	256 6
	row5				25		25			
	row6	2579 7	258 58	2578 578		18		1245 4	1259 19	245 2
	row7		2458	2568 258	158		458	125 12		
	row8	257 7		2578 257	158 15			1257 12		2578 8
	row9	4567 6	458 48		589 89		4589 489		56 5	5678 7

After 5 minutes, duplicate 7's occurred in column 1..

	col 1	col 4	col 9	sep#	result	time
row2	256	26	25		table	1 min
	25	6*	25	sep1	fails	5 min
	6	2*	5*	sep2		

Sudoku #7, continued.

We'll finally test separation 2:

		6			**2**				**5**
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	245 **5**	**1**	**9**	278 **7**	**3**	28 **8**	**6**	25 **2**	47 **4**
►►	row2	256 **6**	**7**	**3**	26 **2**	**4**	**1**	**9**	**8**	25 **5**
	row3	**8**	24 **4**	26 **2**	2679 **6**	**5**	29 **9**	47 **7**	**3**	**1**
	row4	259 **2**	**3**	258 **8**	**4**	18 **1**	**7**	125 **5**	12569 **9**	256 **6**
	row5	**1**	**6**	**4**	25 **5**	**9**	25 **2**	**8**	**7**	**3**
	row6	2579 **9**	258 **5**	2578 **7**	**3**	18 **8**	**6**	1245 **4**	1259 **1**	245 **2**
	row7	**3**	2458 **2**	2568 **6**	158 **8**	**7**	458 **4**	125 **1**	1256 **5**	**9**
	row8	257 **7**	**9**	2578 **5**	158 **1**	**6**	**3**	1257 **2**	**4**	2578 **8**
	row9	4567 **4**	458 **8**	**1**	589 **9**	**2**	4589 **5**	**3**	56 **6**	5678 **7**

Separation 2 succeeds in 7 min. Everything is 9-perfect.

Final Report

	col 1	col 4	col 9	sep#	result	time
row2	256	26	25		table	1 min
	25	6*	25	sep1	fails	5 min
	6	2*	5*	sep2	solves	7 min
					verifies	2 min
					total	15 mins

Sudoku #8

The sudoku below has undergone all the steps of solving - trapping, annotation, row-box reductions, column-box reductions, and group separations. Column 3 is chosen as the basis for the separation analysis.

	col 1	col 2	col 3 ▼	col 4	col 5	col 6	col 7	col 8	col 9
row1	9	1346	2	8	347	146	137	137	5
row2	148	5	18	37	2	14	379	6	379
row3	16	136	7	1356	9	156	8	2	4
row4	3	12	168	28	5	7	69	4	89
row5	78	9	4	36	1	68	2	5	378
row6	578	27	568	9	34	248	367	37	1
row7	1567	167	3	1257	8	9	4	17	27
row8	1457	8	15	12457	6	1245	137	9	237
row9	2	147	9	147	47	3	5	8	6

It took 5 minutes to compute the table below:

Results up to this point:

	col3		gen1		
row2	18	1	8	8	8
row4	168	68*	16*	1	6
row6	568	68*	56*	6*	5*
row8	15	5*	15*	5*	1*
sep#		sep1		sep2	sep3
result	table				
time	5 min				

Sudoku #8, continued.

We first test separation 1:

		▼ col 1	col 2	▼ col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1		1346			347	146	137	137	
		9	4	2	8	3	6	1	7	5
	row2	148		18	37		14	379		379
1		8	5	1	7	2	4	9	6	3
	row3	16	136		1356		156			
		6	3	7	5	9	1	8	2	4
	row4		12	168	28			69		89
68		3	1	8	2	5	7	6	4	8
	row5	78			36		68			378
		7	9	4	3	1	6	2	5	8
	row6	578	27	568		34	248	367	37	
68		5	2	6	9	4	8	7	3	1
	row7	1567	167		1257				17	27
		6	7	3	5	8	9	4	1	2
	row8	1457		15	12457		1245	137		237
5		1	8	5	4	6	2	3	9	7
	row9		147		147	47				
		2	4	9	1	7	3	5	8	6

It took 5 minutes for separation 1 to fail, with duplicate 6's in column 1.

Results up to this point:

	col3		gen2		
row2	18	1	8	8	8
row4	168	68*	16*	1	6
row6	568	68*	56*	6*	5*
row8	15	5*	15*	5*	1*
sep#		sep1		sep2	sep3
result	table	fails			
time	5 min	5 min			

Sudoku #8, continued.

We next test separation 2:

		col 1	col 2	col 3 ▼▼	col 4	col 5	col 6	col 7	col 8	col 9
	row1	9	1346	2	8	347	146	137	137	5
8	row2	148 14	5	18 8	37	2	14	379	6	379
	row3	16	136	7	1356	9	156	8	2	4
1	row4	3	12 2	168 1	28	5	7	69	4	89
	row5	78 8	9	4	36	1	68	2	5	378
6	row6	578 5	27 7	568 6	9	34 4	248	367 7	37 3	1
	row7	1567	167	3	1257	8	9	4	17	27
5	row8	1457	8	15 5	12457	6	1245	137	9	237
	row9	2	147	9	147	47	3	5	8	6

Separation 2 fails in 1 minute with duplicate 7's in row 6.

Results up to this point:

	col3		gen1		
row2	18	1	8	8	8
row4	168	68*	16*	1	6
row6	568	68*	56*	6*	5*
row8	15	5*	15*	5*	1*
sep#		sep1		sep2	sep3
result	table	fails		fails	
time	5 min	5 min		1 min	

Sudoku #8, continued.

We now test separation 3:

		col 1	col 2	▼ col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1		1346			347	146	137	137	
		9	6	2	8	3	4	1	7	5
	row2	148		18	37		14	379		379
8		4	5	8	7	2	1	3	6	9
	row3	16	136		1356		156			
		1	3	7	6	9	5	8	2	4
	row4	3	12	168	28			69		89
6			1	6	2	5	7	9	4	8
	row5	78			36		68			378
		8	9	4	3	1	6	2	5	7
	row6	578	27	568		34	248	367	37	
5		7	2	5	9	4	8	6	3	1
	row7	1567	167		1257				17	27
		6	7	3	5	8	9	4	1	2
	row8	1457		15	12457		1245	137		237
1		5	8	1	4	6	2	7	9	3
	row9		147		147	47				
		2	4	9	1	7	3	5	8	6

Separation 3 succeeds in 8 minutes.

Final Report

	col3		gen1			verify	total
row2	18	1	8	8	8		
row4	168	68*	16*	1	6		
row6	568	68*	56*	6*	5*		
row8	15	5*	15*	5*	1*		
sep#		sep1		sep2	sep3		
result	table	fails		fails	solves		
time	5 min	5 min		1 min	8 min	2 min	21 min

Sudoku #9

The sudoku below has undergone all the steps of solving - trapping, annotation, box-line reductions, and group separations. Row 2 is chosen for separation analysis, as it has only three squares, two with only two candidates.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	4568	9	3	2468	78	267	68	1	578
►► row2	7	1	68	5	9	3	268	26	4
row3	4568	46	2	468	1	67	9	567	3
row4	26	8	17	69	4	5	126	3	279
row5	2346	46	9	1	37	8	5	267	27
row6	36	5	17	369	2	679	168	4	789
row7	89	3	4	289	6	129	7	25	125
row8	1	2	58	7	58	4	3	9	6
row9	69	7	56	239	35	129	4	8	125

Calculation of separations:

	col 1	col 7	col 8	
row 2	68	268	26	
	6	8*	2*	sep1
	8	26*	26	sep2

* = forced choice, underlined = free choice or remaining choice

Sudoku #9, continued.

We shall first test separation1:

6 **8** **2**

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	4568 **5**	**9**	**3**	2468 **4**	78 **8**	267 **2**	68 **6**	**1**	578 **7**
►► row2	**7**	**1**	68 **6**	**5**	**9**	**3**	268 **8**	26 **2**	**4**
row3	4568 **8**	46 **4**	**2**	468 **6**	**1**	67 **7**	**9**	567 **5**	**3**
row4	26 **2**	**8**	17 **7**	69	**4**	**5**	126 **1**	**3**	279 **9**
row5	2346 **4**	46 **6**	**9**	**1**	37 **7**	**8**	**5**	267 **6**	27 **2**
row6	36 **3**	**5**	17 **1**	369 69	**2**	679 69	168 **8**	**4**	789 **7**
row7	89 **9**	**3**	**4**	289	**6**	129	**7**	25	125
row8	**1**	**2**	58 **8**	**7**	58	**4**	**3**	**9**	**6**
row9	69 **6**	**7**	56 **5**	239 29	35 **3**	129	**4**	**8**	125

Separation 1 fails after 3 minutes with triple 69's in box E.

Results up to this point:

	col 1	col 7	col 8		result	time
row 2	68	268	26		table	2 min
	6	8*	2*	sep1	fails	3 min
	8	26*	26	sep2		

Sudoku #9, continued.

We next test separation 2:

				8			**26**		**26**
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	4568 **6**	**9**	**3**	2468 **4**	78 **7**	267 **2**	68 **8**	**1**	578 **5**
►►	row2	**7**	**1**	68 **8**	**5**	**9**	**3**	268 **2**	26 **6**	**4**
	row3	4568 **5**	46 **4**	**2**	468 **8**	**1**	67 **6**	**9**	567 **7**	**3**
	row4	26 **2**	**8**	17 **7**	69 **6**	**4**	**5**	126 **1**	**3**	279 **9**
	row5	2346 **4**	46 **6**	**9**	**1**	37 **3**	**8**	**5**	267 **2**	27 **7**
	row6	36 **3**	**5**	17 **1**	369 **9**	**2**	679 **7**	168 **6**	**4**	789 **8**
	row7	89 **8**	**3**	**4**	289 **2**	**6**	129 **9**	**7**	25 **5**	125 **1**
	row8	**1**	**2**	58 **5**	**7**	58 **8**	**4**	**3**	**9**	**6**
	row9	69 **9**	**7**	56 **6**	239 **3**	35 **5**	129 **1**	**4**	**8**	125 **2**

After 4 minutes, separation 2 results in a solution, a check showing the sudoku to be 9-perfect.

Final Report:

	col 1	col 7	col 8		result	time
row 2	68	268	26		table	2 min
	6	8*	2*	sep1	fails	3 min
gen1	8	26*	26	sep2	solves	4 min
					verify	2 min
					total	11 min

	col 1	col 7	col 8	
winning separation	8	2	6	as can be seen row 2 of the sudoku itself

Sudoku #10

The sudoku below has undergone all the preliminary steps of solving - trapping, annotation, box-line reductions, and group separations. Row 2 is chosen for separation analysis, as it has only three squares, two with only two candidates.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	27	12	127	9	38	5	36	368	4
►► row2	9	5	6	34	2	48	7	38	1
row3	4	8	3	7	1	6	5	2	9
row4	6	234	247	1	347	9	24	5	8
row5	2378	12349	12478	34	5	47	2469	1679	27
row6	5	149	147	8	6	2	49	179	3
row7	1	7	5	2	9	3	8	4	6
row8	238	6	9	5	478	478	1	37	27
row9	238	234	248	6	78	1	239	379	5

Separation Table:

	col 4	col 6	col 8		result	time
row2	34	49	38		table	15 sec
	3	4*	8*	sep1		
	4	9*	3*	sep2		

* = forced choice, underlined = free choice or remaining choice

Sudoku #10, continued.

We first test separation1:

				3		**4**		**8**	
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	27 **7**	12 **1**	127 **2**	**9**	38 **8**	**5**	36 **3**	368 **6**	**4**
►►	row2	**9**	**5**	**6**	34 **3**	**2**	48 **4**	**7**	38 **8**	**1**
	row3	**4**	**8**	**3**	**7**	**1**	**6**	**5**	**2**	**9**
	row4	**6**	247 **2**	247 **7**	**1**	347 **3**	**9**	24 **4**	**5**	**8**
	row5	2378 **3**	12349 **9**	12478 **8**	34 **4**	**5**	47 **7**	2469 **6**	1679 **1**	27 **2**
	row6	**5**	149 **4**	147 **1**	**8**	**6**	**2**	49 **9**	179 **7**	**3**
	row7	**1**	**7**	**5**	**2**	**9**	**3**	**8**	**4**	**6**
	row8	238 **2**	**6**	**9**	**5**	478 **4**	478 **8**	**1**	37 **3**	27 **7**
	row9	238 **8**	234 **3**	248 **4**	**6**	78 **7**	**1**	239 **2**	379 **9**	**5**

Separation 1 solves the sudoku in 3 minutes.

Results:

	col 4	col 6	col 8		result	time
row2	34	49	38		table	15 sec
	3	4*	8*	sep1	solves	3 min
	4	9*	3*	sep2	no test	--------
					verify	2 min
					total	5 min approximately

* = forced choice, underlined = free choice or remaining choice
Note that neither asterisking nor underlining is continued on successive lines, and refer only to actions on the same line

Sudoku #11

The sudoku below has undergone all the preliminary steps of solving - trapping, annotation, box-line reductions, and group separations. Row 2 is chosen for separation analysis, as it has only three squares, two with only two candidates.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	9	23	46	16	1456	56	238	28	7
►► row2	347	8	5	2	47	9	1	6	34
row3	3467	1	2346	3678	4678	367	239	5	2349
row4	15	7	236	9	16	8	235	4	236
row5	15	29	2689	17	3	4	57	289	2689
row6	36	4	3689	5	67	2	3789	1	3689
row7	34	6	349	38	2589	35	289	7	1
row8	8	5	7	4	29	1	6	3	29
row9	2	39	1	3678	6789	367	4	89	5

Separation Table:

	col 1	col 2	col 3		result	time
row 2	347	47	34		table	1 min
	34	7*	34	sep1		
	7	4*	3*	sep2		

Sudoku #11 continued

We first test separation 1:

		34			**7**				**34**
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	9	23	46	16	1456	56	238	28	7
►►	row2	347 **34**	8	5	2	47 **7**	9	1	6	34 34
	row3	3467 7	1	2346	3678	4678	367	239	5	2349
	row4	15	7	236	9	16	8	235	4	236
	row5	15	29	2689	17	3	4	57	289	2689
	row6	36 **6**	4	3689	5	67 **6**	2	3789	1	3689
	row7	**34**	6	349	38	2589	35	289	7	1
	row8	8	5	7	4	29	1	6	3	29
	row9	2	39	1	3678	6789	367	4	89	5

Separation 1 fails in about 10 seconds. The 7 in square (25) makes square (65) = 6, and the 34's in squares (21) and (71) make square (61) = 6. Now there are duplicate 6's in row 6.

Results so far:

	col 1	col 2	col 3		result	time
row 2	347	47	34		table	1 min
	34	7*	34	sep1	fails	10 sec
	7	4*	3*	sep2		

Sudoku #11 continued

So now we carry out separation 2, which we know, by the failure of the combination separation 1, ***must*** be correct, just in order to calculate and verify the result.

		7			**4**				**3**
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	9	23 3	46 4	16 1	1456 5	56 6	238 8	28 2	7
►►	row2	347 7	8	5	2	47 4	9	1	6	34 3
	row3	3467 6	1	2346 2	3678 3	4678 8	367 7	239 9	5	2349 4
	row4	15 5	7	236 6	9	16 1	8	235 3	4	236 2
	row5	15 1	29 2	2689 8	17 7	3	4	57 5	289 9	2689 6
	row6	36 3	4	3689 9	5	67 6	2	3789 7	1	3689 8
	row7	34 4	6	349 3	38 8	2589 9	35 5	289 2	7	1
	row8	8	5	7	4	29 2	1	6	3	29 9
	row9	2	39 9	1	3678 6	6789 7	367 3	4	89 8	5

After 6 minutes, the squares above are all evaluated. A verification shows the resulting grid to be 9-perfect, so separation 3 solves the sudoku.

Final Report:

	col 1	col 2	col 3	sep#	result	time
row 2	347	47	34		table	2 min
	34	7*	34	sep1	fails	negligible
	7	4*	3*	sep2	solves	6 min
					verification	2 min
					total	11 min

Sudoku #12

The sudoku below has undergone all the preliminary steps of solving - trapping, annotation, box-line reductions, and group separations. Row 1 is chosen for separation analysis, as it has only three unsolved squares.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►► row1	1	38	7	5	348	6	9	48	2
row2	238	9	56	237	3478	248	1	56	48
row3	4	28	56	1	28	9	57	567	3
row4	5	348	2	79	79	48	6	38	1
row5	38	7	34	6	1	2458	58	23589	89
row6	6	1	9	23	38	258	4	258	7
row7	7	45	1	8	59	3	2	49	6
row8	23	56	34	29	56	1	78	4789	489
row9	9	26	8	4	26	7	3	1	5

Separation Table

	col 2	col 5	col 8	sep#	result	time
row 1	38	348	48		table	5 min
	<u>38</u>	38*	4*	sep1		
	<u>3</u>	48	48	sep2		
	<u>8</u>	3	4	sep3		

Sudoku #12, continued.

We'll begin with separation 1.

		38			**38**			**4**	
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►►	row1	1	**38** 8	7	5	**348** 3	6	9	**4**8 4	2
	row2	238 3	9	56	237 2	3478 7	248 4	1	56	48 8
	row3	4	28 2	56	1	28 8	9	57	567	3
	row4	5	348 3	2	79	79	48	6	38	1
	row5	38 8	7	34 4	6	1	2458 2	58 5	23589	89 9
	row6	6	1	9	23 2	38 3	258 58	4	258 58	7
	row7	7	45 4	1	8	59 5	3	2	49 9	6
	row8	23 2	56 5	34 3	29 9	56 6	1	78	4789 78	489 4
	row9	9	26 6	8	4	26 2	7	3	1	5

After 2 minutes of updating, duplicate 2's occur in column 4.

Results so far:

	col 2	col 5	col 8	sep#	result	time
row 1	38	348	48		table	5 min
	38	38*	4	sep1	fails	2 min
	3	48	48	sep2	fails	2 min
	8	3	4	sep3		

Sudoku #12, continued.

We'll continue with separation 2:

3 **48** **48**

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►► row1	1	38 3	7	5	348 48	6	9	48	2
row2	238 28	9	56	237 37	3478 37	248	1	56	48
row3	4	28	56	1	28	9	57	567	3
row4	5	348 48	2	79	79	48	6	38	1
row5	38	7	34	6	1	2458	58	23589	89
row6	6	1	9	23	38	258	4	258	7
row7	7	45	1	8	59	3	2	49	6
row8	23	56	34	29	56	1	78	4789	489
row9	9	26	8	4	26	7	3	1	5

After 8 minutes, with very little progress, I could go no further with this temporary update.

Results so far:

	col 2	col 5	col 8	sep#	result	time
row 1	38	348	48		table	5 min
	38	38*	4*	sep1	fails	2 min
	3	48	48	sep2	**bogs down**	2 min+++++
	8	3	4	sep3		

At this point, I'm just going to trudge along with separation 3, which will be investigated on the next page.

Sudoku #12, continued.

Investigation of separation 3:

		8			**3**			**4**	
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►►	row1	1	38 8	7	5	348 3	6	9	48 4	2
	row2	238 3	9	56	237 2	3478 47	248 24	1	56	48 8
	row3	4	28 2	56	1	28 8	9	57	567	3
	row4	5	348 34	2	79 7	79	48	6	38	1
	row5	38	7	34	6	1	2458	58	23589	89 9
	row6	6	1	9	23 3	38 8	258	4	258	7
	row7	7	45	1	8	59 5	3	2	49 9	6
	row8	23	56	34	29 9	56 6	1	78	4789 78	489 4
	row9	9	26	8	4	26 2	7	3	1	5

After 7 minutes, separation 3 fails with duplicate 8's in column 5.

Results so far:

	col 2	col 5	col 8	sep#	result	time
row 1	38	348	48		table	5 min
	38	38*	4*	sep1	fails	2 min
	3	48	48	sep2	**bogs down**	2 min+++++
	8	3	4	sep3	fails	7 min

On the next page, we'll investigate the two halves of sep2.

Sudoku #12, continued.

Investigation of separation 2, FIRST HALF:

		3			**4**			**8**	
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►►	row1	1	38 3	7	5	348 4	6	9	48 8	2
	row2	238 28	9	56	237 37	3478 37	248 28	1	56	48 4
	row3	4	28	56	1	28	9	57	567	3
	row4	5	348 48	2	79	79	48	6	38 3	1
	row5	38	7	34	6	1	2458 245	58	23589 259	89
	row6	6	1	9	23	38	258	4	258 25	7
	row7	7	45	1	8	59	3	2	49	6
	row8	23	56	34	29	56	1	78	4789 479	489 89
	row9	9	26	8	4	26	7	3	1	5

This is as far as I can get with this separation. 2B, the second half of sep1, is known to fail, as it is a specific variation of sep1, which has already failed..

Results so far:

	col 2	col 5	col 8	sep#		result	time
row 1	38	348	48			table	5 min
	38	38*	4*	sep1		fails	2 min
	3	48	48	sep2	combined	bogs down	2 min+++++
	3	4	8*	sep2A	first half	bogs down	2 min+++
	3	8*	4*	sep2B	second half	already shown to fail	

The answer seems to be that sep2A is ***correct***, but ***not extensive enough*** to arrive at a solution. This discussion is continued on the next page.

Sudoku #12B.

Perhaps the row 1 separations didn't fail because they were not ***extensive*** enough. Perhaps they weren't ***effective*** enough. One characteristic I make in choosing a subpattern is the ability of a member of the subpattern to solve its immediate box. It isn't a sure thing, but it is a possible indication of the power of the subpattern. If the same is true for separations, then what we might look for in a separation is its immediate ability to solve the boxes that the elements of the separation exist in.

There is one other row in the sudoku we've been looking at which also has three candidate squares is row 7. Let's look at the number of solved squares it generates in its individual boxes, and compare that to the number of solved squares the row 1 separation generates within its individual boxes. Let's compare ***atomic separations*** – those not containing combination separations.

	col 2	col 5	col 8	
row 1	box A	box B	box C	
separation	3	4	8	
#solved squares	0	0	1	total = 1

	col 2	col 5	col 8	
row 1	box A	box B	box C	
separation	3	8	4	
#solved squares	0	2	1	total = 3

	col 2	col 5	col 8	
row 1	box A	box B	box C	
separation	8	3	4	
#solved squares	2	0	1	total = 3

---

	col 2	col 5	col 8	
row 7	box A	box B	box C	
separation	4	5	9	
#solved squares	4	3	0	total = 7

	col 2	col 5	col 8	
row 7	box A	box B	box C	
separation	5	9	4	
#solved squares	4	3	0	total = 7

Boxes G and H have an interesting property. Look at their elements:

G: 23 34 45 56 62 H: 29 95 56 62

The first digit of each pair is the last digit of the preceding pair. They are ***cyclic boxes***.. Whenever you solve a single pair, you immediately solve all of them. This is a real clue as to why the row 7 separations produce so many more solved squares.

Sudoku #12B, continued.

This time we'll choose row 7 for our separations:

.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	1	38	7	5	348	6	9	48	2
row2	238	9	56	237	3478	248	1	56	48
row3	4	28	56	1	28	9	57	567	3
row4	5	348	2	79	79	48	6	38	1
row5	38	7	34	6	1	2458	58	23589	89
row6	6	1	9	23	38	258	4	258	7
►► row7	7	45	1	8	59	3	2	49	6
row8	23	56	34	29	56	1	78	4789	489
row9	9	26	8	4	26	7	3	1	5

Separation Calculation Table:

	col 2	col 5	col 8	
row 7	45	59	49	
	4	5*	9*	sep1
	5	9*	4*	sep2

Sudoku #12B, continued.

We first try separation 1.

		4			**5**			**9**	
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	1	38 3	7	5	348 4	6	9	48 8	2
	row2	238 8	9	56 5	237 3	3478 7	248 2	1	56 6	48 4
	row3	4	28 2	56 6	1	28 8	9	57 5	567 7	3
	row4	5	348 8	2	79 7	79 9	48 4	6	38 3	1
	row5	38 3	7	34 4	6	1	2458 5	58 8	23589 2	89 9
	row6	6	1	9	23	38 3	258 8	4	258 5	7
►►	row7	7	45 4	1	8	59 5	3	2	49 9	6
	row8	23 2	56 5	34 3	29 9	56 6	1	78 7	4789 4	489 8
	row9	9	26 6	8	4	26 2	7	3	1	5

Final Report:

	col 2	col 5	col 8		result	time
row 7	45	59	49		table	negligible
	4	5*	9*	sep1	solves	7 min
	5	9*	4*	sep2	------	
					verify	2 min
					total	9 min

Sudoku #13

The sudoku below has undergone all the preliminary steps of solving - trapping, annotation, box-line reductions, and group separations. Row 8 is chosen for separation analysis, as it has only three unsolved squares.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	248	5	28	9	47	3	27	6	1
row2	7	3	6	15	2	15	9	4	8
row3	249	1	29	467	467	8	257	3	257
row4	235	9	4	378	578	27	3578	1	6
row5	35	8	357	46	146	14	357	2	9
row6	1	6	2357	378	5789	279	4	78	57
row7	58	7	158	2	1458	145	6	9	3
►► row8	6	2	89	78	3	79	1	5	4
row9	359	4	1359	15	159	6	278	78	27

Separation Table:

	col 3	col 4	col 6	sep#	result	time
row 8	89	78	79		table	1 min
	8	7*	9*	sep1		
	9	8*	7*	sep2		

Sudoku #13, continued.

We'll first test separation 1.

8 7 9

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	8	5	2	9	4	3	7	6	1
row2	7	3	6	1	2	5	9	4	8
row3	4	1	9	6	7	8	2	3	5
row4	2	9	4	8	5	7	3	1	6
row5	3	8	7	4	6	1	5	2	9
row6	1	6	5	3	9	2	4	8	7
row7	5	7	1	2	8	4	6	9	3
►► row8	6	2	89 8	78 7	3	79 9	1	5	4
row9	9	4	3	5	1	6	8	7	2

It took 5 minutes to complete the updating of this sudoku. Verification showed that the sudoku was 9-perfect.

Final Results::

	col 3	col 4	col 6	sep#	result	time
row 8	89	78	79		table	1 min
	8	7*	9*	sep1	solves	5 min
	9	8*	7*	sep2	------	------
					verify	2 min
					total	8 min

Sudoku #14

The sudoku below has undergone all the preliminary steps of solving - trapping, annotation, box-line reductions, and group separations. Row 5 is chosen for separation analysis, as it has only three unsolved squares.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	1	458	458	3	7	568	2	456	9
row2	36	2458	7	256	568	9	146	1345	14
row3	36	9	25	256	4	1	7	356	8
row4	4	13	13	8	56	25	9	7	26
►► row5	5	7	9	46	1	24	3	8	26
row6	28	6	28	7	9	3	14	14	5
row7	7	135	135	456	2	458	1468	9	14
row8	9	248	248	1	3	68	5	26	7
row9	28	15	6	9	58	7	148	124	3

Separation Analysis

	col 4	col 6	col 9	sep#	result	time
row 5	46	24	26		table	30 sec
	4	2*	6*	sep1		
	6	4*	2*	sep2		

Sudoku #14, continued.

We first test separation 1.

				4		**2**			**6**
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	1	458 8	458 4	3	7	568 6	2	456 5	9
	row2	36 3	2458 2	7	256 5	568 8	9	146 6	1345 1	14 4
	row3	36 6	9	25 5	256 2	4	1	7	356 3	8
	row4	4	13 3	13 1	8	56 6	25 5	9	7	26 2
►►	row5	5	7	9	46 4	1	24 2	3	8	26 6
	row6	28 2	6	28 8	7	9	3	14 1	14 4	5
	row7	7	135 5	135 3	456 6	2	458 4	1468 8	9	14 1
	row8	9	248 4	248 2	1	3	68 8	5	26 6	7
	row9	28 8	15 1	6	9	58 5	7	148 4	124 2	3

Separation 1 solves the sudoku in 5 minutes.

Final Report:

	col 4	col 6	col 9	sep#	result	time
row 5	46	24	26		table	30 sec
	4	2*	6*	sep1	solves	5 min
	6	4*	2*	sep2	------	-------
					verify	2 min
					total	8 min

Sudoku #15

This sudoku has been through all preliminary solving techniques, plus group separations, box-line reductions, and some n-wing reductions. Row 7 is chosen for separation analysis.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	4	57	9	8	35	6	2	1	37
row2	2	6	3578	79	1	39	358	4	38
row3	1	58	3578	47	2345	234	3568	67	9
row4	7	124	12	5	249	8	49	3	6
row5	9	3	26	146	246	124	7	8	5
row6	8	45	56	3	469	7	49	2	1
►► row7	5	128	1278	169	368	139	368	67	4
row8	36	9	18	146	7	134	368	5	2
row9	36	78	4	2	368	5	1	9	378

Separation Calculation Table:

	col 2	col 5	col 9		result	time
row 1	57	35	37		table	5 sec
	5	3*	7*	sep1		
	7	5*	3*	sep2		

Sudoku #15, continued.

We'll first test separation 1.

		5			**3**				**7**
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►►	row1	4	57 5	9	8	35 3	6	2	1	37 7
	row2	2	6	3578 3	79 7	1	39 9	358 5	4	38 8
	row3	1	58 8	3578 7	47 4	2345 5	234 2	3568 3	67 6	9
	row4	7	124 1	12 2	5	249 9	8	49 4	3	6
	row5	9	3	26 6	146 1	246 2	124 4	7	8	5
	row6	8	45 4	56 5	3	469 6	7	49 9	2	1
	row7	5	128 2	1278 8	169 9	368 3	139 1	368 6	67 7	4
	row8	36 3	9	18 1	146 6	7	18 4	18 8	5	2
	row9	18 6	18 7	4	2	18 8	5	1	9	18 31

It took 4 minutes to evaluate, plus another 2 minutes to discover the duplicate 3's in column 5.

	col 2	col 5	col 9		result	time
row 1	57	35	357		table	5 sec
	5	3*	7*	sep1	fails	6 min
	7	35	35*	sep2		

Sudoku #15, continued.

We'll next test separation 2.

		7			**5**				**3**
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►►	row1	4	57 7	9	8	35 5	6	2	1	37 3
	row2	2	6	3578 3	79 7	1	39 9	358 5	4	38 8
	row3	1	58 5	3578 8	47 4	2345 2	234 3	3568 6	67 7	9
	row4	7	124 1	12 2	5	249 9	8	49 4	3	6
	row5	9	3	26 6	146 1	246 4	124 2	7	8	5
	row6	8	45 4	56 5	3	469 6	7	49 9	2	1
	row7	5	128 2	1278 7	169 9	368 8	139 1	368 3	67 6	4
	row8	36 3	9	18 1	146 6	7	18 4	18 8	5	2
	row9	18 6	18 8	4	2	18 3	5	1	9	18 7

It took 3 minutes to evaluate, plus 2 minutes to verify.

Final Results:

	col 2	col 5	col 9		result	time
row 1	57	35	37		table	5 sec
	5	3*	7*	sep1	fails	6 min
	7	5*	3*	sep2	solves	3 min
					verify	2 min
					total	11 min

Sudoku #16

This sudoku has been through all preliminary solving techniques, plus group separations, box-line reductions, and some n-wing reductions. Row 4 is chosen for separation analysis.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	1268	9	1268	36	23	7	18	4	5
row2	3	7	16	5	8	4	9	16	2
row3	268	4	5	69	1	29	7	36	38
►► row4	57	8	3	2	45	1	6	9	47
row5	267	26	4	8	39	39	5	12	17
row6	259	1	29	7	45	6	23	8	34
row7	189	23	7	39	6	5	4	123	138
row8	4	236	26	1	7	8	23	5	9
row9	189	5	189	4	239	239	18	7	6

Separation Calculcation Table:

	col 1	col 5	col 9		result	time
row 4	57	45	47		table	2 sec
	5	4*	7*	sep1		
	7	5*	4*	sep2		

Sudoku #16, continued.

We first test separation 1.

		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
					5				**4**					**7**
	row1	1268 **6**	**9**	1268 **8**	36 **3**	23 **2**	**7**	18 **1**	**4**	**5**
	row2	**3**	**7**	16 **1**	**5**	**8**	**4**	**9**	16 **6**	**2**
	row3	268 **2**	**4**	**5**	69 **6**	**1**	29 **9**	**7**	36 **3**	38 **8**
►►	row4	57 **5**	**8**	**3**	**2**	45 **4**	**1**	**6**	**9**	47 **7**
	row5	267 **7**	26 **6**	**4**	**8**	39 **9**	39 **3**	**5**	12 **2**	17 **1**
	row6	259 **9**	**1**	29 **2**	**7**	45 **5**	**6**	23 **3**	**8**	34 **4**
	row7	189 **8**	23 **2**	**7**	39 **9**	**6**	**5**	**4**	123 **1**	138 **3**
	row8	**4**	236 **3**	26 **6**	**1**	**7**	**8**	23 **2**	**5**	**9**
	row9	189 **1**	**5**	189 **9**	**4**	239 **3**	239 **2**	18 **8**	**7**	**6**

It took 4 minutes to evaluate this update, and another 2 minutes to verify that it is the solution.

Final Results:

	col 1	col 5	col 9		result	time
row 4	57	45	47		table	2 sec
	5	4*	7*	sep1	solves	4 min
	7	5*	4*	sep2	------	
					verify	2 min
					total	6 min

Sudoku #17

The sudoku below has undergone trapping, annotation, and preliminary group separation only. It looked immediately ideal for separation analysis. Row 3 was chosen.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	69	2	348	678	5	3678	47	1	39
row2	5	38	348	278	1	9	47	23	6
►► row3	1	69	7	26	4	236	8	5	239
row4	23678	1	238	9	36	247	5	37	48
row5	37	4	9	15	8	15	2	6	37
row6	23678	3678	5	247	36	247	1	9	48
row7	289	89	6	45	7	45	3	28	1
row8	4	78	1	3	2	6	9	78	5
row9	237	5	23	18	9	18	6	4	27

Separation Table:

	col 2	col 4	col 6	col 9		table	1 min
row 3	69	26	236	239			
	6	2*	3*	9*	sep1		
gen1	9	26	236	23*			
	9	2	6*	3*	sep2		
	9	6	23*	23	sep3		

Sudoku #17, continued.

We first test sep1:

		6		**2**		**3**			**9**
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	69 **9**		348 **4**	678 **6**		3678 **8**	47 **7**		39 **3**
	row2		38 **3**	348 **8**	278 **7**			47 **4**	23 **2**	
►►	row3		69 **6**		26 **2**		236 **3**			239 **9**
	row4	23678 **8**		238 **2**		36 **6**	247 **7**		37 **3**	48 **4**
	row5	37 **3**			15 **1**		15 **5**			37 **7**
	row6	23678 **6**	3678 **7**		247 **4**	36 **3**	247 **2**			48 **8**
	row7	289 **2**	89 **9**		45 **5**		45 **4**		28 **8**	
	row8		78 **8**						78 **7**	
	row9	237 **7**		23 **3**	18 **8**		18 **1**			27 **2**

Separation 1 solved the sudoku in 3 min

Final Report:

	col 2	col 4	col 6	col 9		result	time
row 3	69	26	236	239		table	1 min
	6	2*	3*	9*	sep1	solved	3 min
gen1	9	26	236	23*			
	9	2	6*	3*	sep2	--------	
	9	6	23*	23	sep3	--------	
						verify	2 min
						total	6 min

Sudoku #18

The sudoku below has undergone all the steps of solving, trapping, annotation, box-line reductions, and group separations. Row 6 is chosen for separation analysis, as it has only 4 unsolved squares, two with only two candidates each..

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	238	1	235	39	6	25	2459	7	458
row2	9	567	23567	8	37	4	256	25	1
row3	2678	5678	4	19	179	25	3	2589	568
row4	48	3	1	7	2	9	458	6	458
row5	5	6789	679	14	148	3	4789	489	2
►► row6	478	2	79	5	48	6	4789	1	3
row7	236	569	8	3469	349	7	1	245	456
row8	1	567	3567	2	34	8	4567	45	9
row9	267	4	2679	69	5	1	2678	3	678

Separation Table:

col 1	col 3	col 5	col 7		table	1 min
478	79	48	4789			
4	79	8*	79*	separation 1	fails	2 min
7	9*	48	48	separation 2		
8	79	4*	79*	separation 3		

* = forced choice, underlined = free choice

Sudoku #18, continued.

We shall first test separation1:

		4		**79**		**8**		**79**		
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	238 3	1	235	39 9	6	25	2459 4	7	458 8
	row2	9	567	23567	8	37 3	4	256 26	25	1
	row3	2678 **26**	5678 8	4	19 1	179 7	25	3	2589 9	568 56
	row4	48 8	3	1	7	2	9	458 5	6	458 45
	row5	5	6789 679	679	14	148 14	3	4789 789	489 89	2
►►	row6	478 4	2	79	5	48 8	6	4789 79	1	3
	row7	236 **26**	569	8	3469	349	7	1	245	456
	row8	1	567	3567	2	34	8	4567 67	45	9
	row9	267 **26**	4	2679 269	69	5	1	2678 268	3	678 7

After 2 minutes, separation 1 fails with three 26's in column 1.

col 1	col 3	col 5	col 7		table	1 min
478	79	48	4789			
4	79	8*	79*	separation 1	fails	2 min
7	9*	48	48	separation 2		
8	79	4*	79*	separation 3		

* = forced choice, underlined = free choice, except for last

Sudoku #18, continued.

We shall next test separation 2.

		7		**9**		**48**		**48**		
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	238 **8**	**1**	235 **5**	39 **3**	**6**	25 **2**	2459 **9**	**7**	458 **4**
	row2	**9**	567 **6**	23567 **3**	**8**	37 **7**	**4**	256 **2**	25 **5**	**1**
	row3	2678 **2**	5678 **7**	**4**	19 **1**	179 **9**	25 **5**	**3**	2589 **8**	568 **6**
	row4	48 **4**	**3**	**1**	**7**	**2**	**9**	458 **5**	**6**	458 **1**
	row5	**5**	6789 **8**	679 **6**	14 **4**	148 **1**	**3**	4789 **7**	489 **9**	**2**
►►	row6	478 **7**	**2**	79 **9**	**5**	48 **8**	**6**	4789 **4**	**1**	**3**
	row7	236 **3**	569 **9**	**8**	3469 **6**	349 **4**	**7**	**1**	245 **2**	456 **5**
	row8	**1**	567 **5**	3567 **7**	**2**	34 **3**	**8**	4567 **6**	45 **4**	**9**
	row9	267 **6**	**4**	2679 **2**	69 **9**	**5**	**1**	2678 **8**	**3**	678 **7**

After 7 minutes, the sudoku was completely solved. Another 2 minutes of counting from 1 to 9 – twenty seven times – showed it to be 9-perfect

Final Results:

	col 1	col 3	col 5	col 7		table	1 min
row 6	478	79	48	4789			
	4	79	8*	79*	separation 1	fails	2 min
	7	9*	48	48	separation 2	solves	7 min
	8	79	4*	79*	separation 3	-------	
						verify	2 min
						total	12 min

* = forced choice, underlined = free choice

Sudoku #19

The sudoku below has gone through all the steps of solving, including trapping, annotation, box-line reductions, and group separations. Row 1 is arbitrarily chosen for separation analysis.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►► row1	9	78	678	146	2	147	3	18	5
row2	57	4	1	8	57	3	6	9	2
row3	5678	3	2	9	567	157	17	4	178
row4	478	1	3	245	9	245	2457	6	478
row5	2	5	48	7	3	6	14	18	9
row6	467	9	467	1245	8	1245	12457	3	147
row7	148	2	489	56	456	458	149	7	3
row8	147	6	5	3	47	9	8	2	14
row9	3	78	4789	24	1	2478	49	5	6

	col 2	col 3	col 4	col 6	col 8	
	78	678	146	147	18	
gen1	7	68*	146	14*	18	
gen2	7	6	14	14	8*	.
	7	6	1	4*	8	sep1
	7	6	4	1*	8	sep2
	7	8	6*	4*	1*	sep3
gen3	8	7*	46	46	1*	
	8	7	4	6*	1	sep4
	8	7	6	4*	1	sep5

* = forced choice, underlined = open choice (7 minutes to calculate this table)

Sudoku #19, continued.

We shall first test separation1:

7 6 1 4 8

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►► row1	9	78 7	678 6	146 1	2	147 4	3	18 8	5
row2	57 5	4	1	8	57 7	3	6	9	2
row3	5678 8	3	2	9	567 6	157 5	17	4	178 17
row4	478 4	1	3	245	9	245	2457	6	478
row5	2	5	48 8	7	3	6	14	18	9
row6	467 6	9	467 7	1245	8	1245	12457	3	147
row7	148 1	2	489 49	56	456 5	458	149 49	7	3
row8	147 7	6	5	3	47 4	9	8	2	14
row9	3	78 8	4789 49	24	1	2478	49	5	6

Failed, due to the fatal 4 in squares (73), (93), (77) and (97)
It took me 5 min to test this separation.

Sudoku #19, continued.

We next test separation 2:

7 6 4 1 8

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►► row1	9	78 7	678 6	146 4	2	147 1	3	18 8	5
row2	57 5	4	1	8	57 7	3	6	9	2
row3	5678 8	3	2	9	567 6	157 5	17 7	4	178 1
row4	478 7	1	3	245 5	9	245 4	2457 2	6	478 8
row5	2	5	48 8	7	3	6	14 4	18 1	9
row6	467 6	9	467 4	1245 1	8	1245 2	12457 5	3	147 7
row7	148 4	2	489 9	56 6	456 5	458 8	149 1	7	3
row8	147 1	6	5	3	47 4	9	8	2	14 1
row9	3	78 8	4789 7	24 2	1	2478 7	49 9	5	6

It took me 4 minutes to arrive at this contradiction
Subpattern 2 fails with duplicate 7s in row 9.

(Note again that different routes of updating will generally arrive at different contradictions when the separation is not correct, but that all routes of updating will arrive at exactly the same conclusion when the separation is the correct one.)

Sudoku #19, continued.

We next test separation3:

7 8 6 4 1

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►► row1	9	78 7	678 8	146 6	2	147 4	3	18 1	5
row2	57 5	4	1	8	57 7	3	6	9	2
row3	5678 6	3	2	9	567 5	157 1	17 7	4	178 8
row4	478 8	1	3	245 4	9	245 5	2457	6	478
row5	2	5	48 4	7	3	6	14	18	9
row6	467 7	9	467 6	1245 1	8	1245 2	12457	3	147
row7	148 1	2	489 9	56 5	456 6	458 8	149	7	3
row8	147 4	6	5	3	47 7	9	8	2	14
row9	3	78 8	4789 7	24 4	1	2478 7	49	5	6

It took me 3 minutes to arrive at this point.
Separation 3 fails because of the duplicate 4s in column 4.

Sudoku #19, continued.

We next test separation 4:

8 6 4 7 1

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►► row1	9	78 8	678 6	146 4	2	147 7	3	18 1	5
row2	57 7	4	1	8	57 5	3	6	9	2
row3	5678 5	3	2	9	567 6	157 1	17 7	4	178 8
row4	478 8	1	3	245 5	9	245 4	2457 2	6	478 7
row5	2	5	48 4	7	3	6	14 1	18 8	9
row6	467 6	9	467 7	1245 1	8	1245 2	12457 5	3	147 4
row7	148 1	2	489 8	56 6	456 4	458 5	149 9	7	3
row8	147 4	6	5	3	47 7	9	8	2	14 1
row9	3	78 7	4789 9	24 2	1	2478 8	49 4	5	6

It took me 5 minutes to complete this update.
All rows, columns, and boxes are 9-perfect, so separation 4 is the solution.

See the next page for the final statistics.

Sudoku #19, continued.

Final Results

	col 2	col 3	col 4	col 6	col 8	sep#	result	time
	78	678	146	147	18		table	7 minutes
gen1	7	68*	146	14*	18			
gen2	7	6	14	14	8*			
	7	6	1	4*	8	sep1	fails	5 minutes
	7	6	4	1*	8	sep2	fails	2 minutes
	7	8	6*	4*	1*	sep3	fails	3 minutes
gen3	8	7*	46	46	1*			
	8	7	4	6*	1	sep4	solves	5 minutes
	8	7	6	4*	1	sep5	-------	

total time 22 minutes

* = forced choice, underlined = open choice, gen1, 2 & 3 are general lines calculated to facilitate subsequent analysis of the separations.

Sudoku #20

The sudoku below has only undergone trapping, annotation, and group separations. Row 1 is chosen for candidate separation analysis.

		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►►	row1	3	9	478	25678	4567	1	24	47	2457
	row2	6	5	147	27	9	247	124	8	3
	row3	18	78	2	578	3	4578	6	9	1457
	row4	4	2678	578	139	57	23579	138	167	178
	row5	9	1	3	67	8	467	5	2	47
	row6	258	2678	578	13	457	23457	1348	1467	9
	row7	158	38	6	39	2	3589	7	14	148
	row8	7	4	589	568	1	568	289	3	268
	row9	128	238	189	4	67	3678	189	5	168

	col 3	col 4	col 5	col 7	col 8	col 9			
row 1	478	25678	4567	24	47	2457		table	12 min
	4	8*	6*	2*	7*	5*	sep1		
	7	8*	6*	2*	4*	5*	sep2		
gen1	8	2567	4567	24	47	2457			
	8	2	6*	4*	7*	5*	sep3		
gen2	8	5	467	24	47	247			
imp1	8	5	**4**	2*	7*	----			
gen3	8	5	6	24	47	247			
	8	5	6	2	47	47*	sep4		
	8	5	6	4	7*	2*	sep5		

* = forced choice, underlined = open choice,
gen1, 2 & 3 are general lines calculated to facilitate subsequent analysis
imp1 is an impossible line included to explain why underscored choice is skipped.

Sudoku #20, continued.

Testing sep1.

				4	**8**	**6**		**2**	**7**	**5**
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
►►	row1			478	25678	4567		24	47	2457
		3	9	4	8	6	1	2	7	5
	row2			147	27		247	124		
		6	5	1	2	9	7	4	8	3
	row3	18	78		578		4578			1457
		8	7	2	5	3	4	6	9	1
	row4		2678	578	139	57	23579	138	167	178
		4	2	8	1	5	9	3	6	7
	row5				67		467			47
		9	1	3	7	8	6	5	2	4
	row6	258	2678	578	13	457	23457	1348	1467	
		5	6	7	3	4	2	8	1	9
	row7	158	38		39		3589		14	148
		1	3	6	9	2	5	7	4	8
	row8			589	568		568	289		268
		7	4	5	6	1	8	9	3	2
	row9	128	238	189		67	3678	189		168
		2	8	9	4	7	3	1	5	6

Separation 1 solved the sudoku in 7 minutes. It took another 2 minutes to verify 9-completeness.

Sudoku #20, continued.

Final Results:

	col 3	col 4	col 5	col 7	col 8	col 9		result	time
row 1	478	25678	4567	24	47	2457		table	12 min
	4	8*	6*	2*	7*	5*	sep1	solves	7 min
	7	8*	6*	2*	4*	5*	sep2	-------	
gen1	8	2567	4567	24	47	2457			
	8	2	6*	4*	7*	5*	sep3	-------	
gen2	8	5	467	24	47	247			
imp1	8	5	**4**	2*	7*	----			
gen3	8	5	6	24	47	247			
	8	5	6	2	47	47*	sep4	-------	
	8	5	6	4	7*	2*	sep5	-------	
								verify	2 min
								total	21 min

Needless to say, this might have taken 49 minutes if sep5 had solved the sudoku.

By way of comparison, using the 3 pattern, it took about the same time to calculate the subpatterns as it did to calculate the separation table (there are 5 subpatterns, in comparison to 5 separations), and the fifth subpattern actually took 9 minutes to solve the sudoku. If we had been lucky, and tested the fifth subpattern first, it would have taken 9 minutes to solve the sudoku with it, so solving by subpattern would have taken 23 minutes altogether, so the two methods seem, in this example at least, about the same. The upshot seems to indicate that solving by separation takes slightly less time than solving by subpattern. Perhaps if I improved my grasp of standard methods, I wouldn't need to use either one.

I have no idea how much fast the average solver would solve this sudoku using standard methods , since I only succeed in solving about half the sudokus I undertake without solving by subpattern.. That's the only reason I began solving by subpattern – I wasn't successful at solving many sudokus without it.

Sudoku #21

The sudoku below has gone through all the steps of solving - ttrapping, annotation, box-line reductions, and group separations. Row 7 is chosen for separation analysis because it looks promising – small number of candidates (4), within a sudoku not having large numbers of candidates in its squares

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	346	34	9	1	7	2	8	3456	345
row2	367	5	178	489	389	489	136	2	39
row3	2	348	148	6	389	5	134	349	7
row4	579	279	3	589	6	1789	4	589	2589
row5	8	279	57	359	4	79	236	3569	1
row6	459	1	6	3589	2	189	7	3589	3589
►► row7	1	79	478	2	89	3	5	48	6
row8	359	6	58	489	1	489	23	7	238
row9	34	348	2	7	5	6	9	1	348

Separation Table:

	col 2	col 3	col 5	col 8		result	time
row 7	79	478	89	48		table	2 min
	7	48*	9*	48	sep1		
	9	7*	8*	4*	sep2		

* = forced choice, underlined = open choice (7 minutes to calculate this table)

Sudoku #21, continued.

We first test subpattern 1.

		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
			7	**48**		**9**			**48**	
	row1	346 6	34	 9	 1	 7	 2	 8	3456 345	345
	row2	367 7	 5	178 18	489	389	489	136 6	 2	39
	row3	 2	348	148	 6	389	 5	134 1	349	 7
	row4	579 5	279 9	 3	589 8	 6	1789	 4	589 5	2589 2
	row5	 8	279 2	57 7	359 5	 4	79 9	236 3	3569 6	 1
	row6	459 4	 1	 6	3589	 2	189	 7	3589	3589
►►	row7	 1	79 7	478 48	 2	89 9	 3	 5	48	 6
	row8	359 9	 6	58 5	489	 1	489	23	 7	238
	row9	34 3	348 48	 2	 7	 5	 6	 9	 1	348

After 4 minutes, separation 1 failed with duplicate 5's in row 4.

	col 2	col 3	col 5	col 8		result	time
row 7	79	478	89	48		table	2 min
	7	48*	9*	48	sep1	failed	4 min
	9	7*	8*	4*	sep2		

Sudoku #21, continued.

We next test subpattern 2

		9	**7**		**8**			**4**	
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	346 **6**	34 **3**	**9**	**1**	**7**	**2**	**8**	3456 **5**	345 **4**
	row2	367 **7**	**5**	178 **1**	489 **8**	389 **3**	489 **4**	136 **6**	**2**	39 **9**
	row3	**2**	348 **8**	148 **4**	**6**	389 **9**	**5**	134 **1**	349 **3**	**7**
	row4	579 **9**	279 **7**	**3**	589 **5**	**6**	1789 **1**	**4**	589 **8**	2589 **2**
	row5	**8**	279 **2**	57 **5**	359 **9**	**4**	79 **7**	236 **3**	3569 **6**	**1**
	row6	459 **4**	**1**	**6**	3589 **3**	**2**	189 **8**	**7**	3589 **9**	3589 **5**
►►	row7	**1**	79 **9**	478 **7**	**2**	89 **8**	**3**	**5**	48 **4**	**6**
	row8	359 **5**	**6**	58 **8**	489 **4**	**1**	489 **9**	23 **2**	**7**	238 **3**
	row9	34 **3**	348 **4**	**2**	**7**	**5**	**6**	**9**	**1**	348 **8**

.

It took 6 minutes for separation 2 to solve the sudoku, and another 2 minutes to verify that it is 9-perfect.

Final Report:

	col 2	col 3	col 5	col 8		result	time
row 7	79	478	89	48		table	2 min
	7	48*	9*	48	sep1	failed	4 min
	9	7*	8*	4*	sep2	solved	6 min
						verify	2 min
						total	14 min

Sudoku #21, continued.

I also solved this sudoku using a simple subpattern

		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1						X			
	row2								X	
	row3	X								
	row4		2_1							2_2
	row5		2_2					2_1		
	row6					X				
►►	row7				X					
	row8							2_2		2_1
	row9			X						

Subpattern-1 failed & subpattern-2 succeeded in times comparable to the solution by candidate separation. There were 2 subpatterns and 2 separations, and the times for each were virtually the same as for the solution by separation. There are many parallels between the two methods, because they are related. But the logic behind their use is the real reason for their similarity.

Sudoku #22

For a change, we'll do the separation using a box. The following sudoku has been through all the usual steps of trapping, annotation, box-line reductions, and group separations. Box B is chosen for separation analysis.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	56	7	9	2	1	4	3	68	568
row2	56	1	4	**678**	3	**678**	679	2	569
row3	8	3	2	9	5	**67**	1	67	4
row4	49	58	6	4578	2	1	4789	4789	3
row5	7	2	3	48	6	89	489	5	1
row6	49	58	1	3	49	5789	2	46789	689
row7	3	49	5	16	49	2	68	168	7
row8	12	6	7	145	8	59	49	3	29
row9	12	49	8	146	7	3	5	1469	269

For convenience in creating the separation table, we'll show this box as a simple horizontal 9-string, with the designations of the squares shown immediately above the members of the string.:The separation table took less than half a minute to calculate.

Separation Table:

	(24)	(26)	(36)		result	time
box B	678	678	67		table	30 sec
	6	8*	7*	sep1		
	7	8*	6*	sep2		
	8	67	67	sep3		

Sudoku #22, continued.

We'll first test separation 1:

	(24)	(26)	(36)	
box B	678	678	67	
	6	8*	7*	sep1

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	56	7	9	2	1	4	3	68	568
row2	56	1	4	678 **6**	3	678 **8**	679	2	569
row3	8	3	2	9	5	67 **7**	1	67	4
row4	49	58	6	4578	2	1	4789	4789	3
row5	7	2	3	48	6	89 9	489	5	1
row6	49	58	1	3	49	5789 5	2	46789	689
row7	3	49	5	16	49	2	68	168	7
row8	12	6	7	145	8	59 9	49	3	29
row9	12	49	8	146	7	3	5	1469	269

This update failed in about 20 seconds, updating column 6 alone, producing two 9's in that column:

	(24)	(26)	(36)			
box B	678	678	67		table	30 sec
	6	8*	7*	sep1	fails	20 sec

Sudoku #22, continued.

We'll next test separation 2:

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	56 **6**	**7**	**9**	**2**	**1**	**4**	**3**	68 **8**	568 **5**
row2	56 **5**	**1**	**4**	678 **7**	**3**	678 **8**	679 **6**	**2**	569 **9**
row3	**8**	**3**	**2**	**9**	**5**	67 **6**	**1**	67 **7**	**4**
row4	49 **4**	58 **8**	**6**	4578 **5**	**2**	**1**	4789 **7**	4789 **9**	**3**
row5	**7**	**2**	**3**	48 **8**	**6**	89 **9**	489 **4**	**5**	**1**
row6	49 **9**	58 **5**	**1**	**3**	49 **4**	5789 **7**	**2**	46789 **6**	689 **8**
row7	**3**	49 **4**	**5**	16 **6**	49 **9**	**2**	68 **8**	168 **1**	**7**
row8	12 **1**	**6**	**7**	146 **4**	**8**	59 **5**	49 **9**	**3**	29 **2**
row9	12 **2**	49 **9**	**8**	146 **1**	**7**	**3**	**5**	1469 **4**	269 **6**

Separation 2 succeeded in solving this sudoku in 5 minutes.

Final Report:

	(24)	(26)	(36)		result	time
box B	678	678	67		table	30 sec
	6	8*	7*	sep1	fails	20 sec
	7	8*	6*	sep2	solves	5 min
	8	67	67	sep3	-------	
					verify	2 min
					total	8 min

Sudoku #23

The sudoku below went through trapping, annotation, and group separations, but no n-wings. Row 2 is chosen for candidate separation analysis.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	3	4	5	2	9	67	67	8	1
row2	9	6	27	13	18	138	27	5	4
►► row3	12	127	8	45	67	45	9	267	3
row4	1246	127	347	8	156	9	34567	1467	57
row5	5	1789	79	146	3	146	678	167	2
row6	146	18	34	7	156	2	34568	146	9
row7	24	29	6	359	78	3678	1	247	57
row8	8	5	149	19	2	17	47	3	6
row9	7	3	12	156	4	156	25	9	8

Separation Analysis:

	col 1	col 2	col 4	col 5	col 6	col 8		result	time
row 3	12	127	45	67	45	267		table	10 min
gen1	1	27*	45	67	45	267			
	1	2	45	67	45	67*	sep1		
	1	7	45	6*	45	2*	sep2		
gen2	2	17*	45	67	45	67			
	2	1	45	67	45	67	sep3		
imp	2	7	45	6*	45	---			

imp represents an impossible line, included to make it plain why the choice which made it impossible did not lead to a separation. The imp line is not a necessary one.

Sudoku #23, continued.

We'll first test separation1.

		1	**2**		**45**	**67**	**45**		**67**	
		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
	row1	3	4	5	2	9	67 6	67 7	8	1
	row2	9	6	27 7	13 3	18 1	138 8	27 2	5	4
►►	row3	12 1	127 2	8	45 4	67 7	45 5	9	267 6	3
	row4	1246 2	127 7	347 4	8	156 6	9	34567 3	1467 1	57 5
	row5	5	1789 8	79 9	146 1	3	146 4	678 6	167 7	2
	row6	146 6	18 1	34 3	7	156 5	2	34568 8	146 4	9
	row7	24 4	29 9	6	359 5	78 8	378 3	1	247 2	57 7
	row8	8	5	149 1	17 9	2	17 7	47 4	3	6
	row9	7	3	12 2	156 6	4	156 1	25 5	9	8

After 5 minutes, separation 1 solves the sudoku.

Final Report:

	col 1	col 2	col 4	col 5	col 6	col 8		result	time
row 3	12	127	45	67	45	267		table	10 min
gen1	1	27*	45	67	45	267			
	1	2	45	67	45	67*	sep1	solves	5 min
	1	7	45	6*	45	2*	sep2	------	
gen2	2	17*	45	67	45	67			
	2	1	45	67	45	67	sep3	------	
								verify	2 min
								total	17 min

Note that the algorithm does not discover the triple combination

12	12	45	67	45	67	

which is a combination of sep1 & sep3, and also solves the sudoku.

Sudoku #24

This sudoku has been through all the initial solving techniques. Row 4 is selected for separation analysis.

	col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
row1	238	268	9	35678	5678	23578	1	458	45
row2	238	5	4	138	18	238	9	6	7
row3	7	68	1	9	568	4	2	58	3
►► row4	159	147	6	57	3	579	8	1479	2
row5	589	78	2	4	56789	1	3	79	69
row6	189	1478	3	678	2	789	5	1479	469
row7	4	3	8	2	59	6	7	59	1
row8	6	9	7	1358	158	358	4	2	58
row9	12	12	5	78	4	789	6	3	89

Separation Calculation Table:

	col 1	col 2	col 4	col 6	col 8		result	time
row 3	159	147	57	579	1479		table	10 min
	1	4	5	79*	79*	sep1		
	1	4	7	5*	9*	sep2		
	1	7	5*	9*	4*	sep3		
	5	14	7*	9*	14	sep4		
imp	5	7	--	9*	14			
	9	14	57	57	14	sep5		
imp	9	7	5*	--	14			

Sudoku #24, continued.

We'll first test separation 1:

		col 1	col 2	col 3	col 4	col 5	col 6	col 7	col 8	col 9
		1	**4**		**5**		**79**		**79**	
	row1	238 8	268 2	9	35678 7	5678 6	23578 3	1	458 4	45 5
	row2	238 3	5	4	138 1	18 8	238 2	9	6	7
	row3	7	68 6	1	9	568 5	4	2	58 8	3
►►	row4	159 1	147 4	6	57 5	3	579 9	8	1479 7	2
	row5	589 5	78 8	2	4	56789 7	1	3	79 9	69 6
	row6	189 9	1478 7	3	678 6	2	789 8	5	1479 1	469 4
	row7	4	3	8	2	59 9	6	7	59 5	1
	row8	6	9	7	1358 3	158 1	358 5	4	2	58 8
	row9	12 2	12 1	5	78 8	4	789 7	6	3	89 9

Separation 1 solved the sudoku in 5 minutes. A verification revealed the sudoku to be 9-perfect.

Final Report:

	col 1	col 2	col 4	col 6	col 8		result	time
row 3	159	147	57	579	1479		table	10 min
	1	4	5	79*	79*	sep1	solves	5 min
	1	4	7	5*	9*	sep2	------	
	1	7	5*	9*	4*	sep3	------	
	5	14	7*	9*	14	sep4	------	
	9	14	57	57	14	sep5	------	
							verify	2 min
							total	17 min

Combined versus Atomic Separations.

As a prelude to the next section, which is devoted to problems in calculating separations, it is useful to go into some of the peculiar relationships of the two styles of separation, the first being a mixture of atomic and combined separations, and the second being composed of atomic separations only. The term ***atomic*** refers to a separation involving no combined separations, being composed of single digits only, and a combined separation being partially composed of pairs of digits and single digits.

Comparison of combined versus atomic:

The sudoku involved is Sudoku #11, where we shall consider the separation based on column 6 (rather than row 2): We'll show it in both forms, first using combined separations whenever possible, and the second using atomic separations only.

The three conceivable combinations are pairs of 36's, pairs of 67's and pairs of 37's. The outcomes are shown as well, since we know them all from the solution of Sudoku #11:

Mixed form:

	row 1	row 3	row 7	row 9		result
col 6	56	367	35	367		
imp	5	36	---	36		
	5	67	3*	67	sep1	fails
	6	37	5*	37	sep2	solves

Atomic form:

	row 1	row 3	row 7	row 9		result
col 6	56	367	35	367		
gen1	5	67*	3*	67*		
	5	6	3*	7*	sep1A	fails
	5	7	3*	6*	sep1B	fails
gen2	6	37*	5*	37*		
	6	3	5*	7*	sep2A	solves
	6	7	5*	3*	sep2B	fails

The only peculiarity here is the labeling of the four separations in the atomic form as sep1A, sep1B, sep2A, and sep 2B, instead of the traditional sep1, sep2, sep3, and sep4. We have done this in order to better compare the two sets of results. Note that sep1A is the first half of the combined sep1, sep1B the second half of the combined sep1, sep2A the first half of the combined sep2, and sep2B the second half of the combined sep2, each combined separation being broken down into two separate atomic separations, referred to as the two halves of the combined separation.

These results are entirely in line with the earlier section, **Some Remarks on Combined Separations**. Nothing here is new. This is just a reiteration of the principles stated in that section, applied to a particular separation. Whenever you perform a separation, you will need to choose one or the other of these two forms, the mixed or the atomic.

Combined versus Atomic Separations, continued.

My experience with these two forms is that they both work, and that a solver is at liberty to choose either one, but it is useful to point out the differences.

The first is that, in this example at least, there are only two combined separations to test versus four atomic separations. The second is that the atomic separation is easier, perhaps, to understand, and the testing is slightly easier, even though there's more of it. The information in either case is the same. Even though you might think that the first form doesn't tell you which half worked, it actually does. You need only look at the solution to see. Where there's a pair of digits in the separation itself, there's only one digit in the solution to the sudoku, so you can see at a glance which half did the solving.

This doesn't mean that combined separations are always best. Let's look at another separation, this time of column 5 of Sudoku #10 (instead of row 2):

Second comparison of combined versus atomic:

Mixed form:

	row 1	row 4	row 8	row 9		result
col 5	38	347	478	78		
	3	47*	47*	8*	sepA	fails
	3	4	78*	78*	sepB	fails
	3	7	4*	8*	sepC	fails
	8	3*	4*	7*	sepD	solves

Atomic form:

	row 1	row 4	row 8	row 9		result
col 5	38	347	478	78		
gen1	3	47*	478	78		
gen2	3	4	78*	78		
	3	4	7	8*	sep1	fails
	3	4	8	7*	sep2	fails
	3	7	4*	8*	sep3	fails
	8	3*	4*	7*	sep4	solves

The first thing that strikes one is that both methods have four separations to test. Clearly, in this case the atomic method is easier to understand. The relations between the two methods are a bit peculiar:

sepA = combination of sep1 and sep3
sepB = combination of sep1 and sep2
sepC = sep3
sepD = sep4

Combined and Atomic Separations, continued.

These peculiarities do not pose a problem, but they are not aesthetic. Clearly the atomic form is preferable in this situation. This particular example poses an interesting question:: Could a similar situation exist in another sudoku where sep1 was the solution? If so, then both sepA and sepB would solve the sudoku, since a combined separation only needs one of its halves to be correct in order to solve the sudoku. Both solved sudokus would be identical in appearance, so there would be no paradox. The only difference would be the possible paths the two solutions could follow in updating the sudoku. But just getting two solutions in itself would be a bit startling. However, since we normally stop testing when we have a solution, we'd probably never even notice it.

Although the preceding example seems to suggest that atomic separations are preferable to mixed separations, which include combined spearations, the following example shows otherwise.

	col A	col B	col C	col D	col E	col F	
	24	1249	458	458	129	289	
gen1	2	149*	458	458	19*	89*	
	2	1	45*	45*	9*	8*	sep1
	2	4	58*	58*	1*	9*	sep2
	2	9	45*	45*	1*	8*	sep3
	4	1	58*	58*	9*	2*	sep4
	4	1	58*	58*	2*	9*	sep5
gen2	4	2	58*	58*	19*	89*	
	4	2	58*	58*	1*	9*	sep6
	4	9	58	58	1*	2*	sep7

Moreover, sep4 & sep5 could be combined into a single separation

4	1	58	58	29	29	sepC

and sep6 & sep7 could also be combined into another single separation

4	29	58	58	1	29	sepD

giving us only five separations to test: sep1, sep2, sep3, sepC, and sepD.

These new combinations were not discovered by my proferred method of calculating separations, so this seems to be an imperfection in that method (but the non-combinations, the atomic ones, do cover the same ground). Still, it is more important that the method not miss any separations and that it be easy to follow. Combinations can always be made after the fact if they are not automatically discovered.

On the next page, for the same problem, we'll calculate atomic separations only.

The same problem, with atomic separations only:

	col A	col B	col C	col D	col E	col G	
	24	1249	458	458	129	289	
gen1	2	149*	458	458	19*	89*	
gen1A	2	1	45*	45*	9*	8*	
	2	1	4	5*	9	8	sep1
	2	1	5	4*	9	8	sep2
imp	2	1	8	45	9*	--	
gen1B	2	4	58	58	1*	9*	
	2	4	5	8*	1	9	sep3
	2	4	8	5*	1	9	sep4
gen1C	2	9	45*	45*	1*	8*	
	2	9	4	5*	1	8	sep5
	2	9	5	4*	1	8	sep6
gen2	4	129*	58*	58*	129	289	
gen2A	4	1	58	58	29*	29*	
gen2B	4	1	5	8*	29	29	
	4	1	5	8	2	9*	sep7
	4	1	5	8	9	2*	sep8
gen2C	4	1	8	5*	29	29	
	4	1	8	5	2	9*	sep9
	4	1	8	5	9	2*	sep10
gen3	4	2	58*	58*	19*	89*	
	4	2	5	8*	1*	9*	sep11
	4	2	8	5*	1	9	sep12
gen3A	4	9	58*	58*	12*	28*	
gen3B	4	9	58	58	1*	2*	
	4	9	5	8*	1	2	sep13
	4	9	8	5*	1	2	sep14

---

14 atomic separations versus 5 combined separations. Which would you choose? Note again that combined separations generally seem to work as well as atomic separations. This even holds sometimes for triple combinations. There might be occasional combined separations which don't completely solve their associated sudokus while the atomic ones do, but they are rare (in my somewhat brief experience)..

Atomic separation by example.

	col A	col B	col C	col D
row x	38	347	478	78

1. We begin a new line, selecting the leftmost candidate of the leftmost column, underscoring it, and then removing it from every other column it appears in:

	col A	col B	col C	col D
row x	38	347	478	78
	3	47*	478	78

Note that when we did this, we asterisked the candidate list for every column that the underscored candidate was removed from.

If the resultant line is composed only of single candidates, we label it on the right as a separation line; otherwise we label it on the left as a general line. In this case, it is a general line, and it is the first general line, so we label it gen1 (note that the original list of candidates is really a general line itself. It just isn't labeled as such)..

	col A	col B	col C	col D
row x	38	347	478	78
gen1	3	47*	478	78

Since the resultant line is not composed of single digits, we start another line, selecting the leftmost candidate of the second column (col B), which is a 4, and we underline it. Then we remove it from every group of candidates for each of the columns.

	col A	col B	col C	col D
row x	38	347	478	78
gen1	3	47*	478	78
	3	4	78*	78

If the resultant line is composed of single candidates, we label it as a separation line; otherwise we label it as a general line, which in this case it is:

	col A	col B	col C	col D
row x	38	347	478	78
gen1	3	47*	478	78
gen1A	3	4	78*	78

Notice that whenever we start a new line, we omit the underlines and asterisks from the preceding line, then insert new underlines and asterists on the new line whenever appropriate.

Since the last line was a general line, we begin a new line, selecting the leftmost candidate, and remove it from all subsequent columns, asterisking them whenever appropriate.

Atomic separation, continued.

	col A	col B	col C	col D	
row x	38	347	478	78	
gen1	3	47*	478	78	
gen1A	3	4	78*	78	
	3	4	7	8*	sep1

Since this last line is composed of single candidates, it is a separation line, and it is the first separation, so we label it sep1

We now return to the most recent general line, which is gen1A, and we copy it as the new last line, omitting the general label, as well as any underscores or asterists.

	col A	col B	col C	col D	
row x	38	347	478	78	
gen1	3	47*	478	78	
gen1A	3	4	78*	78	
	3	4	7	8*	sep1
	3	4	78	78	

If this line has candidates not yet selected, we select the leftmost, as before:

	col A	col B	col C	col D	
row x	38	347	478	78	
gen1	3	47*	478	78	
gen1A	3	4	78*	78	
	3	4	7	8*	sep1
	3	4	8	7*	

If this new line is composed of single candidates only, we label it as a separation line, which, in this case it is:

	col A	col B	col C	col D	
row x	38	347	478	78	
gen1	3	47*	478	78	
gen1A	3	4	78*	78	
	3	4	7	8*	sep1
	3	4	8	7*	sep2

We now return to the previous general line, and discover that it has no further candidates to select, so we go to the previous general line that still has unselected candidates That would be gen1, which still has a 7 under col B. We start a new line as a copy of it, leaving out its label, and omitting all underlines and asterisks. See the next page.

Atomic separation, continued.

	col A	col B	col C	col D	
row x	38	347	478	78	
gen1	3	47*	478	78	
gen1A	3	4	78*	78	
	3	4	7	8*	sep1
	3	4	8	7*	sep2
	3	47	478	78	

Col B has a yet-unselected candidate, namely 7, which we must begin a selection on.

	col A	col B	col C	col D	
row x	38	347	478	78	
gen1	3	47*	478	78	
gen1A	3	4	78*	78	
	3	4	7	8*	sep1
	3	4	8	7*	sep2
	3	7	4*	8*	sep3

Note that two actions occurred with this selection. Col B was reduced to a 48, and col D was reduced to an 8, which then reduced col C to a 4. Since the resultant line is composed of single candidates only, it must be labeled as a separation line, which we also do.

When we return to the most previous general line, there is still an unselected digit, the 8 in column A, so we must begin another line, created from the original column entries. Since this line has an unselected digit, it must be labeled as a general line, and the 8 must be selected.

	col A	col B	col C	col D	
row x	38	347	478	78	
gen1	3	47*	478	78	
gen1A	3	4	78*	78	
	3	4	7	8*	sep1
	3	4	8	7*	sep2
	3	7	4*	8*	sep3
gen2	8	347	47*	7*	

The 7 in col D reduces col C to a 4, which reduces col B to a 3.

Atomic separation, continued.

	col A	col B	col C	col D	
row x	38	347	478	78	
gen1	3	47*	478	78	
gen1A	3	4	78*	78	
	3	4	7	8*	sep1
	3	4	8	7*	sep2
	3	7	4*	8*	sep3
gen2	8	347	47*	7*	
	8	3*	4*	7*	sep4

This new added line is composed of single candidates, so it is a separation. There are no additional candidates to select, so we are done..

Example of mixed separation.

	col A	col B	col C	col D		
row x	38	347	478	78		
gen1	3	47*	478	78		
	3	4	78	78	sep1	(combination)
	3	7	4*	8*	sep2	(atomic)
gen2	8	347	47*	7*		
	8	3*	4*	7*	sep3	(atomic)

You might wonder where the following combination separation is???

	3	**47**	**47**	**8**	
Half of it	3	4	7	8	is hidden in sep1
and the other	3	7	4	8	is sep2 itself.

There is no point in listing it as well, as nothing is added.

That is because the two combination separations

	3	4	78	78
and	3	47	47	8

are overlapping.

Separation analysis: my standard system.

The standard system always makes selections from left to right. No attempt should be made to guarantee pairs of doubles. They will either fall out naturally, or will be allowed to be found in the form of atomic separations. The important rule is to stick to the rules, so no separation will be missed. Using the standard system the pairs of doubles most often found only in their atomic form are those close to the left, where most of the selection takes place.

The most important aspect of analyzing separations is to not miss one, because that one might be the one that solves the sudoku. For this reason it is important to master the art of turning over every stone. The first principle is to have an orderly system. The only system I know is the following:

The selection logic progresses from left to right. The updating of each row is done in every direction, keeping every candidate group in mind at the same time. Consider the following row:

col A	col B	col C	col D	col E
127	45	247	457	1457

Notice that 1 is a candidate in column groups A & E.. The important rule to keep in mind is that the number of unique candidates is always equal to the number of candidate groups. No candidate can be lost. The candidate 1 is only in two columns, A and E. This means that if 1 is not selected as the choice for group A, it must be chosen for group E. Only one candidate is selected for each group to form a separation. That means that if 1 is not selected for column A, it must become the sole survivor for column E, and all other candidates in E are deleted.

Let's begin by selecting the 1 candidate for group A. We place an underscore under it and jettison the other candidates. Note that in doing so, we rejected the 2 candidate, and only one other group, C, has a 2 candidate. Therefore the candidate chosen to represent group C must be that 2 candidate. Now our row looks like this:

col A	col B	col C	col D	col E
1	45	2*	457	457

We'll next select the 4 for group B. Now the choices are forced:

col A	col B	col C	col D	col E	
1	4	2*	57	57	sep1

Note that we came to an end, with a pair of doubles in the last two columns. We could go further, and break this down into two atomic separations, but the standard system accepts all naturally occurring double pairs. This is to keep the separations, wherever possible, to a minimum, since every separation requires testing. At the same time, we don't vary our sequence of actions just to score a double pair, because we might thereby miss one.

	col A	col B	col C	col D	col E	
	127	45	247	457	1457	
gen1	1	45	2*	457	457	
	1	4	2*	57	57	sep1
gem2	27	45	247	457	1457	

Notice that if we next chose the 2 for column 1, there would be only three candidates for four columns. There is only one other column with a 2, namely column C, which must therefore be the choice for that column. This leaves only a 7 for column 1:

	col A	col B	col C	col D	col E	
	127	45	247	457	1457	
gen1	1	45	2*	457	457	
	1	4	2*	57	57	sep1
gen2	7	45	2*	45	145	

Now, the only column left with a 1 is E, which must therefore claim the 1 for itself. Except for the pair of 45's in columns B & D, all columns are reduced to single digits, so we now have our second separation:

	col A	col B	col C	col D	col E	
	127	45	247	457	1457	
gen1	1	45	2*	457	457	
	1	4	2*	57	57	sep1
	7	45	2*	45	1*	sep2

There are no further possibilities, because there is nothing left to select, except, perhaps, to break the combined separation with the pair of 45's into two atomic separations. We are done.

If the number of separations unveiled by the standard system is large, it might be desirable to combine atomic separations, wherever possible, into combination separations. This should be done only at the end, after all separations have been found, because to opt for doubles while using the standard system can confuse the logical order and result in undiscovered atomic separations. This decision is, naturally, one that you as a solver must make for yourself, and should be done only when you have become adept at finding all the separations. There are situations, as shown earlier, when the use of pairs of doubles is desirable in reducing the amount of testing, but there are subtle situations when a preoccupation with finding all the doubles results in an atomic separation being missed.

Problems using the Standard System.

1. Begin all separations with single selections.
2. Accept all combination separations when they occur naturally.

The answers to the following problems are shown in the answers section on page 94.

Problem 1:	56	68	568	
Problem 2:	678	678	67	
Problem 3:	467	46	47	
Problem 4:	16	36	12	123
Problem 5:	78	37	58	35
Problem 6:	23	37	28	78
Problem 7	36	78	368	378
Problem 8:	56	367	35	367
Problem 9:	79	478	89	48
Problem 10.	237	34	27	47
Problem 11:	248	28	47	27
Problem 12:	124	12	249	49
Problem 13:	78	36	68	378
Problem 14:	69	26	236	239
Problem 15:	36	78	368	378
Problem 16:	18	168	568	15
Problem 17:	26	567	267	25
Problem 18:	78	36	68	378
Problem 19:	18	168	568	15
Problem 20:	38	347	478	78
Problem 21:	136	13	236	23
Problem 22:	57	567	456	47
Problem 23:	478	48	467	467
Problem 24:	147	27	247	124
Problem 25:	24	29	25	459
Problem 26	36	78	368	378
Problem 27.	24	349	39	234
Problem 28.	24	349	39	234
Problem 29:	256	257	267	26
Problem 30:	359	56	369	569
Problem 31:	149	147	49	179
Problem 32:	1235	15	35	12
Problem 33:	346	34	3456	345
Problem 34:	146	14	1468	148
Problem 35:	123	123	39	239
Problem 36:	458	458	568	456
Problem 37:	248	248	68	26
Problem 38:	256	257	267	26
Problem 39:	178	478	147	14
Problem 40:	26	146	246	124
Problem 41:	234	247	347	24

Problem 42.	18	258	158	128		
Problem 43.	38	47	17	34	138	
Problem 44.	348	79	79	48	38	
Problem 45.	28	15	58	148	124	
Problem 46.	259	29	45	23	34	
Problem 47.	24	26	2679	29	47	
Problem 48.	268	69	29	36	38	
Problem 49.	268	69	29	36	38	
Problem 50:	379	89	378	27	237	
Problem 52:	36	18	146	134	368	
Problem 53:	28	56	28	57	567	
Problem 54	12	49	146	1469	269	
Problem 55:	678	67	89	5789	59	
Problem 56.	5678	567	157	17	178	
Problem 57.	257	278	25	358	23	
Problem 58.	267	2679	69	2678	678	
Problem 59:	567	23567	37	256	25	
Problem 60:	89	289	129	25	125	
Problem 61:	678	67	89	5789	59	
Problem 62:	568	569	689	29	269	
Problem 63:	68	168	49	29	1469	269
Problem 64:	358	3568	49	49	368	368
Problem 65:	678	4578	48	16	145	146
Problem 66:	346	367	579	459	359	34

Answers to Problems

Problem 1:	col A	col B	col C	
row x	56	68	568	
	5	68	68	sep1
	6	8*	5*	sep2

Problem 2:	col A	col B	col C	
row x	678	678	67	
	6	8*	7*	sep1
	7	8*	6*	sep2
	8	67*	67	sep3

Problem 3:	col A	col B	col C	
row x	467	46	47	
	4	6*	7*	sep1
	6	4*	7*	sep2
	7	6*	4*	sep3

Problem 4:	col A	col B	col C	col D	
row x	16	36	12	123	
	1	6*	2*	3*	sep1
	6	3*	12	12	sep2

Problem 5:	col A	col B	col C	col D	
row x	78	37	58	35	
	7	3*	8*	5*	sep1
	8	7*	5*	3*	sep2

Problem 6:	col A	col B	col C	col D	
row x	23	37	28	78	
	2	3*	8*	7*	sep1
	3	7*	2*	8*	sep2

Problem 7:	col A	col B	col C	col D	
row x	36	78	368	378	
gen1	3	78	68*	78*	
	3	78	6*	78	sep1
	6	78	3*	78	sep2

Problem 8:	col A	col B	col C	col D	
row x	56	367	35	367	
	5	67*	3*	67*	sep1
	6	37*	5*	37*	sep2

Problem 9:	col A	col B	col C	col D	
	79	478	89	48	
gen1	7	48*	89	48	
	7	48	9*	48	sep1
	9	7*	8*	4*	sep2

Problem 10.	col A	col B	col C	col D	
row x	237	34	27	47	
	2	3*	7*	4*	sep1
	3	4*	2*	7*	sep2
	7	3*	2*	4*	sep3

Problem 11:	col A	col B	col C	col D	
row x	248	28	47	27	
	2	8*	4*	7*	sep1
	4	8*	7*	2*	sep2
	8	2*	4*	7*	sep3

Problem 12:	col A	col B	col C	col D	
row x	124	12	249	49	
	1	2*	49*	49	sep1
	2	1*	49*	49	sep2
	4	1*	2*	9*	sep3

Problem 13:	col A	col B	col C	col D	
row x	78	36	68	378	
gen1	7	36	68	38	
	7	3	6*	8*	sep1
	7	6	8*	3*	sep2
	8	3*	6*	7*	sep3

Problem 14	col A	col B	col C	col D	
row x	69	26	236	239	
	6	2*	3*	9*	sep1
gen1	9	26	236	23*	
	9	2	6*	3*	sep2
	9	6	23	23	sep3

Problem 15:	col A	col B	col C	col D	
row x	36	78	368	378	
gen1	3	78	68*	78*	
	3	78	6*	78	sep1
gen2	6	78	38*	378	
	6	7	38	38*	sep2
	6	8	3*	7*	sep3

Problem 16:	col A	col B	col C	col D	
row x	18	168	568	15	
	1	68*	68*	5*	sep1
gen1	8	16	56	15	
	8	1	6*	5*	sep2
	8	6	5*	1*	sep3

Problem 17:	col A	col B	col C	col D	
row x	26	567	267	25	
	2	67*	67*	5*	sep1
gen1	6	57	27	25	
	6	5	7*	2*	sep2
	6	7	2*	5*	sep3

Problem 18:	col A	col B	col C	col D	
row x	78	36	68	378	
	7	3	6*	8*	sep1
	7	6	8*	3*	sep2
	8	3*	6*	7*	sep3

Problem 19:	col A	col B	col C	col D	
row x	18	168	568	15	
	1	68*	68*	5*	sep1
gen1	8	16*	56*	15	
	8	1	6*	5*	sep2
	8	6	5*	1*	sep3

Problem 20:	col A	col B	col C	col D	
row x	38	347	478	78	
gen1	3	47	478	78	
	3	4	78*	78	sep1
	3	7	4*	8*	sep2
	8	3*	4*	7*	sep3
imp	8	4	7*	--	
imp	8	7	4*	--	

Problem 21:	col A	col B	col C	col D	
row x	136	13	236	23	
	1	3*	6*	2*	sep1
	3	1*	6*	2*	sep2
	6	1*	23*	23	sep3
imp	6	3*	2*	--	

Problem 22:	col A	col B	col C	col D	
row x	57	567	456	47	
gen1	5	67*	46*	47	
	5	6	4*	7*	sep1
	5	7	6*	4*	sep2
	7	56*	56*	4*	sep3

Problem 23:	col A	col B	col C	col D	
row x	478	48	467	467	
	4	8*	67	67	sep1
gen1	7	48	46	46	
	7	8*	46	46	sep2
	8	4*	67	67	sep3

Problem 24	col A	col B	col C	col D	
row x	147	27	247	124	
gen1	1	2	47*	4*	
	1	2	7*	4	sep1
	4	27	27	1*	sep2
	7	2*	4*	1*	sep3

Problem 25	col A	col B	col C	col D	
row x	24	29	25	459	
	2	9*	5*	4*	sep1
	4	29	25	59*	
	4	2	5*	9*	sep2
	4	9	2*	5*	sep3

Problem 26	col A	col B	col C	col D	
row x	36	78	368	378	
	3	78	6*	78	sep1
gen2	6	78	38*	378	
	6	7	38	38*	sep2
	6	8	3*	7*	sep3

Problem 27.	col A	col B	col C	col D	
row x	24	349	39	234	
	2	3	9*	4*	sep1
	2	4	9*	3*	sep2
	2	9	3*	4*	sep3
	4	39*	39	2*	sep4

Problem 28.	col A	col B	col C	col D	
row x	24	349	39	234	
	2	3	9*	4*	sep1
	2	4	9*	3*	sep2
	2	9	3*	4*	sep3
	4	39	39	2*	sep4

Problem 29:		col A	col B	col C	col D	
row x		256	257	267	26	
		2	5*	7*	6*	sep1
	gen1	5	27*	267	26	
		5	2	7*	6*	sep2
		5	7	26*	26	sep3
		6	5*	7*	2*	sep4
	imp	6	7	2	--	

Problem 30:	col A	col B	col C	col D	
row x	359	56	369	569	
	3	5	69*	69*	sep1
	3	6	9*	5*	sep2
	5	6*	3*	9*	sep3
	9	56	3*	56*	sep4

Problem 31:		col A	col B	col C	col D	
row x		149	147	49	179	
	gen1	1	47	49	79	
		1	4	9*	7*	sep1
		1	7	4*	9*	sep2
		4	17*	9*	17*	sep3
		9	17*	4*	17*	sep4

Problem 32	col A	col B	col C	col D	
row x	1235	15	35	12	
	1	5*	3*	2*	sep1
	2	5*	3*	1*	sep2
	3	1*	5*	2*	sep3
	5	1*	3*	2*	sep4

Problem 33:	col A	col B	col C	col D	
row x	346	34	3456	345	
	3	4*	6*	5*	sep1
	4	3*	6*	5*	sep2
gen1	6	34	345*	345	
	6	3	45*	45*	sep3
	6	4	35*	35*	sep4

Problem 34:	col A	col B	col C	col D	
row x	146	14	1468	148	
	1	4*	6*	8*	sep1
	4	1*	6*	8*	sep2
	6	1	48*	48*	sep3
	6	4	18*	18*	sep4

Problem 35:	col A	col B	col C	col D	
row x	123	123	39	239	
	1	2	39	39	sep1
	1	3	9*	2*	sep2
	2	1	39	39	sep3
imp	2	3	9*	---	
	3	1	9*	2*	sep4
imp	3	2	9*	---	

Problem 36:	col A	col B	col C	col D	
row x	458	458	568	456	
gen1	4	58*	568	56*	
	4	5	8*	6*	sep1
	4	8	56*	56	sep2
gen2	5	48*	68*	46*	
	5	4	8*	6*	sep3
	5	8	6*	4*	sep4

Problem 37:	col A	col B	col C	col D	
row x	248	248	68	26	
	2	4*	8*	6*	sep1
gen1	4	28*	68	26	
	4	2	8*	6*	sep2
	4	8	6*	2*	sep3
gen2	8	24*	6*	2*	
	8	4*	6	2	sep4

Problem 38:	col A	col B	col C	col D	
row x	256	257	267	26	
	2	5*	7*	6	sep1
gen2	5	27*	267	26	
	5	2	7*	6*	sep2
	5	7	26	26	sep3
gen3	6	257	27*	2*	
	6	5*	7*	2*	sep4

Problem 39:	col A	col B	col C	col D	
row x	178	478	147	14	
	1	8*	7*	4*	sep1
gen1	7	48*	14*	14	
	7	8*	14	14	sep2
gen2	8	47*	147	14	
	8	4	7*	1*	sep3
	8	7	14*	14	sep4

Problem 40:	col A	col B	col C	col D	
row x	26	146	246	124	
gen1	2	146	46*	14*	
	2	1	6*	4*	sep1
	2	4	6*	1*	sep2
	2	6	4*	1*	sep3
gen2	6	14	24	124	
	6	1	24	24	sep4
	6	4	2*	1*	sep5

Problem 41:	col A	col B	col C	col D	
row x	234	247	347	24	
	2	7*	3*	4*	sep1
gen1	3	247	47*	24	
	3	2	7*	4*	sep2
	3	4	7*	2*	sep3
	3	7	4*	2*	sep4
	4	7*	3*	2*	sep5

Problem 42.	col A	col B	col C	col D	
row x	18	258	158	128	
gen1	1	258	58*	28*	
	1	2	5*	8*	sep1
	1	5	8*	2*	sep2
	1	8	5*	2*	sep3
	8	2	5*	1*	sep4
	8	5	1*	2*	sep5

Problem 43.	col A	col B	col C	col D	col E	
row x	38	47	17	34	138	
	3	7*	1*	4*	8*	sep1
gen1	8	47	17	34	13*	
	8	4	7*	3*	1*	sep2
	8	7	1*	4*	3*	sep3

Problem 44.	col A	col B	col C	col D	col E	
row x	348	79	79	48	38	
	3	79	79	4*	8*	sep1
	4	79	79	8*	3*	sep2
	8	79	79	4*	3*	sep3

Problem 45.	col A	col B	col C	col D	col E	
row x	28	15	58	148	124	
gen1	2	15	58	148	14*	
	2	1	5*	8*	4*	sep1
	2	5	8*	14	14	sep2
	8	1	5*	4*	2*	sep3
imp	8	5	---	14	124	

Problem 46.	col A	col B	col C	col D	col E	
row x	259	29	45	23	34	
	2	9*	5*	3*	4*	sep1
	5	9*	4*	2*	3*	sep2
	9	2*	5*	3*	4*	sep3

Problem 47.	col A	col B	col C	col D	col E	
row x	24	26	2679	29	47	
	2	6*	7*	9*	4*	sep1
gen1	4	26	269	29	7*	
	4	2	6*	9*	7*	sep2
	4	6	29*	29	7*	sep3

Problem 48.	col A	col B	col C	col D	col E	
row x	268	69	29	36	38	
	2	6*	9*	3*	8*	sep1
	6	9*	2*	3*	8*	sep2
	8	9*	2*	6*	3*	sep3

Problem 49.	col A	col B	col C	col D	col E	
row x	268	69	29	36	38	
	2	6*	9*	3*	8*	sep1
	6	9*	2*	3*	8*	sep2
	8	9*	2*	6*	3*	sep3

Problem 50:	col A	col B	col C	col D	col E	
row x	3578	79	39	358	38	
	3	7*	9*	5*	8*	sep1
	5	7*	9*	38*	38	sep2
	7	9*	3*	5*	8*	sep3
	8	7*	9*	5*	3*	sep4

Problem 51.	col A	col B	col C	col D	col E	
row x	379	89	378	27	237	
gen1	3	89	78	27	27	
	3	9*	8*	27	27	sep1
gen2	7	89	38*	2*	3*	
	7	9*	8*	2	3	sep2
gen3	9	8*	37*	27	237	
	9	8	3	27	27*	sep3
	9	8	7	2*	3*	sep4

Problem 52:	col A	col B	col C	col D	col E	
row x	36	18	146	134	368	
gen1	3	18	146	14	68*	
gen1A	3	1	46	4*	68	
	3	1	6*	4*	8*	sep1
	3	8	14*	14	6*	sep2
gen2	6	18	14*	134	38*	
	6	1	4*	3*	8*	sep3
	6	8	14*	14*	3*	sep4

Problem 53:	col A	col B	col C	col D	col E	
row x	28	56	28	57	567	
	2	5	8*	7*	6*	sep1
	2	6	8*	57	57	sep2
	8	5	2*	7*	6*	sep3
	8	6	2*	57	57	sep4

Problem 54	col A	col B	col C	col D	col E	
row x	12	49	146	1469	269	
gen1	1	49	46	469	2*	
	1	4	6*	9*	2	sep1
	1	9	46	46*	2	sep2
	2	4	1*	69*	69*	sep3
	2	9	14*	14*	6*	sep4

Problem 55:	col A	col B	col C	col D	col E	
row x	678	67	89	5789	59	
gen1	6	7*	89	589	59	
	6	7	8	59*	59	sep1
gen2	7	6*	89	589*	59	
	7	6	8	59*	59	sep2
gen3	7	6*	9	58*	5*	
	7	6	9	8*	5	sep3
	8	6	9*	7*	5*	sep4
	8	7	--	5*	9*	

Problem 56.	col A	col B	col C	col D	col E	
row x	5678	567	157	17	178	
	5	6*	17	17	8*	sep1
gen1	6	57	157	17	8*	
	6	5	17*	17	8	sep2
gen2	6	7	15	1*	8	
	6	7	5*	1	8	sep3
gen3	7	56*	15	1*	8*	
	7	6*	5*	1	8	sep4
imp	8	5*	17	17	17	

Problem 57.	col A	col B	col C	col D	col E	
row x	257	278	25	358	23	
	2	7*	5*	8*	3*	sep1
	5	7*	2*	8*	3*	sep2
gen1	7	28*	25*	358	23	
	7	2	5*	8*	3*	sep3
gen2	7	8	25	35	23	
	7	8	2	5*	3*	sep4
	7	8	5	3*	2*	sep5

Problem 58.	col A	col B	col C	col D	col E	
row x	267	2679	69	2678	678	
gen1	2	679	69	678	678	
	2	6	9*	78	78	sep1
	2	7	9*	68	68	sep2
	2	9	6*	78	78	sep3
gen2	6	79*	9*	278	78	
	6	7*	9*	2*	8*	sep4
gen3	7	69	69	28	8*	
	7	69*	69	2*	8*	sep5

Problem 59:	col A	col B	col C	col D	col E	
row x	567	23567	37	256	25	
	5	37*	37	6*	2*	sep1
	6	37*	37	25*	25	sep2
gen1	7	2356*	3*	256	25	
	7	2	3	6*	5*	sep3
	7	3	--	256	25	

Problem 60:	col A	col B	col C	col D	col E	
row x	89	289	129	25	125	
gen1	8	29	129	25	125	
gen1A	8	2	19*	5*	1*	
	8	2	9*	5*	1*	sep1
gen2	8	9	12*	25	125	
	8	9	1	25	25	sep2
	8	9	2	5*	1*	sep3
gen3	9	28*	12*	25	125	
imp	9	2	1*	5*	----	
gen4	9	8	12	25	125	
	9	8	1	25	25*	sep4
	9	8	2	5*	1*	sep5

Problem 61:	col A	col B	col C	col D	col E	
row x	678	67	89	5789	59	
	6	7*	89	589*	59	
	6	7	8	59	59	sep1
	6	7	9	8*	5*	sep2
gen1	7	6*	89	589*	59	
	7	6	8	5*	9*	sep3
	7	6	9	8*	5*	sep4
gen2	8	67	9*	57*	5*	
	8	6	9	7*	5	sep5
imp	8	7	--	5*	9*	

Problem 62:	col A	col B	col C	col D	col E	
row x	568	569	689	29	269	
gen1	5	69*	8*	29	269	
	5	6	8	29	29	sep1
	5	9	8	2*	6*	sep2
gen2	6	5*	89*	29	29	
	6	5	8*	29	29	sep3
gen3	8	5*	69	29	269	
	8	5	6	29	29*	sep4
	8	5	9	2*	6*	sep5

Problem 63	col A	col B	col C	col D	col E	col F	
row x	68	168	49	29	1469	269	
gen1	6	8*	49	29	149*	29*	
	6	8	4	29	1*	29	sep1
imp	6	8	9	2	1	---	
gen2	8	1	49	29	46	269	
	8	1	4	29	6*	29	sep2
	8	6	4	29	1*	29	sep3

Problem 63:	col A	col B	col C	col D	col E	col F	
row x	358	3568	49	49	368	368	
	3	5*	49	49	68	68	sep1
gen1	5	368	49	49	368	368	
	5	3	49	49	68*	68*	sep2
	5	6	49	49	38*	38*	sep3
	5	8	49	49	36*	36*	sep4
imp	8	3	49	49	6*	6*	
	8	5	49	49	36	36	sep5
imp	8	6	49	49	3*	3*	

Problem 64:	col A	col B	col C	col D	col E	col F	
row x	678	4578	48	16	145	146	
	6	7*	8*	1*	5*	4*	sep1
gen1	7	458	48	16	145	146	
	7	4	8*	16	5*	16	sep2
	7	5	8	6	14	14	sep3
	7	5	8*	16	4*	16	sep4
	7	8	4*	16	5*	16	sep5

Problem 65:	col A	col B	col C	col D	col E	col F	
row x	346	367	579	459	359	34	
	3	6*	7*	59	59	4*	sep1
	4	6*	7*	59	59	3*	sep2
	6	3	7*	59	59	4*	sep3
	6	7	5	9	3	4	sep4
	6	7*	59	4*	59	3*	sep5

Problem 66:	col A	col B	col C	col D	col E	col F	
row x	346	367	579	459	359	34	
	3	6*	7*	59	59	4*	sep1
	4	6*	7*	59	59	3*	sep2
gen1	6	37	579	459	359	34	
	6	3	7*	59	59	4*	sep3
gen2	6	7	59	459	359	34	
gen3	6	7	5	49*	39*	34*	
	6	7	5	4*	9*	3*	sep4
	6	7	5	9*	3*	4*	sep5
gen4	6	7	9	45	35	34	
	6	7	9	4*	5*	3*	sep6
	6	7	9	5*	3*	4*	sep7

APPENDIX

Box-line elimination

Definition. The term ***line*** will be used to indicate either ***row*** or ***column***. Therefore the term ***box-line*** means either ***box-row*** or ***box-column***.

Box-row elimination theorem.:

If two or three squares are in the same row and in the same box, and those squares possess the same candidate, then

a. if no other square in the row has that candidate, then no other square in the box may have that candidate, ***and***

b. if no other square in the box has that candidate, then no other square in the row may have that candidate.

Proof: Let the candidate be x, and suppose that no other square in the row has that candidate x. Then suppose that another square of the box, not in that row, also has that candidate x. Then imagine that candidate x being promoted to become the established (only) value for its square. It must then be eliminated as a candidate from all the other squares of the box, including the two or three squares originally specified as possessing that candidate. But then no square in the row would possess that particular candidate. This contradicts the requirement that every row of a sudoku possess every value, for if a row does not possess a particular candidate x, then it has no candidate which can be promoted to the value **X**. In the following diagram, that candidate is x, and when it is promoted to the established value **X**, it is moved to the center of its square and printed in large font. It then eliminates all candidates with that value x from the box, resulting in the row no longer possessing x as a candidate:

x	x	x						
x								

The columnar corollary of this theorem is also true.

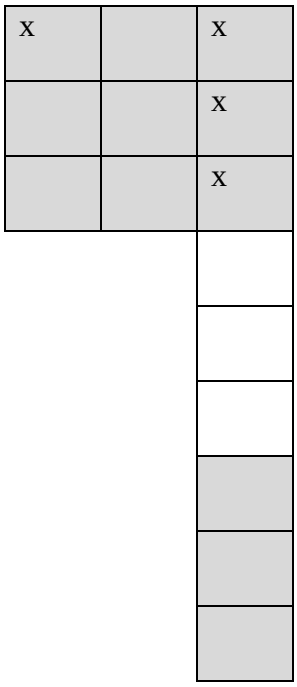
Box-column elimination theorem:

If two or three squares are in the same column and in the same box, and those squares possess the same candidate, then

1. if no other square in the column has that candidate, then no other square in the box may have that candidate, and

2. if no other square in the box has that candidate, then no other square in the column may have that candidate.

The proof of this is similar, requiring only a rotation of the above proof through an angle of 90 degrees and the replacement of “row” by “column”:



This theorem, in its combined form, will be known simply as the **Box-line Theorem**, and is an important tool in the reduction of values of a sudoku. It should be one of the standard situations looked for by the solver during the process of reduction. An example of its use is shown on the next page. Note that the three squares specified in the diagram could just as easily be two squares, as the principle is the same.

The box-line principle should be used during the initial reduction of candidates immediately after annotation, but the box-line situation is sometimes more evident during the search for n-wings.

On the next page, we shall examine we examine the relationship between the bottom box and the leftmost column in different situations. Each of these stacks represents the left stack (column 1, column 2 & column 3) in ***different***, although very similar-looking, sudokus:

Situation A.

	col 1	col 2	col 3
row1	135	4	58
row2	135	6	7
row3	12	1289	289
row4	9	1257	3
row5	126	12	26
row6	8	257	4
row7	257	25~~7~~9	259
row8	2467	25~~7~~8	268
row9	247	3	1

Situation B.

	col 1	col 2	col 3
row1	135	4	58
row2	135	7	6
row3	125	1289	2589
row4	9	5	3
row5	126~~7~~	126	27
row6	8	26	4
row7	257	29	259
row8	2467	28	289
row9	247	3	1

Situation C.

	col 1	col 2	col 3
row1	135	4	58
row2	135	6	7
row3	125	1289	2589
row4	9	5	3
row5	1267	127	26
row6	8	27	4
row7	257	279	259
row8	2467	278	268
row9	247	3	1

In **situation A**, squares (71), (81) & (91) of box [31] are the **only** squares in column 1 of the sudoku containing the candidate 7. They are therefore the **only** squares in box [31] which can have 7 as a candidate, for if one of the other squares of box [31] had a candidate 7, and if that candidate were to become the established 7 for its square, it would become the only 7 in box [31], causing the candidate 7 in squares (71), (81) & (91) to be eliminated and there would then be no 7's in column 1 of the sudoku.. Therefore we can eliminate the candidate 7's from squares (72) and (82).

In **situation B**, there are no other candidate 7's in box [31] besides those in squares (71), (81) & (91), so the leftmost three squares in column 1 are the only squares in column 1 which can have 7 as a candidate. Therefore the candidate 7 may be erased from square (51). In **situation C**, the candidate 7's in squares (71), (81) & (91) are not unique in either their column or box, and no eliminations can be made of the candidate 7.

Nomenclature

(For the naming conventions for squares, rows, columns, boxes, bands, and stacks, glance over Chapter 1 of the Subpattern Analysis Textbook.)

(For an understanding of "separation of values" consult the Separation of Values text.)

The ***true value*** of a square is the digit value that it inherently possesses, generally unknown at the beginning of solving, when the square is blank, but which will be determined by the solver during the process of solving the sudoku. At the beginning, before solving, the ***true values*** of a small number of squares, usually from 24 to 27 (never less than 18, and sometimes as great as 35). These are known as ***givens***, . They are the clues to solving the puzzle. All of the blank squares have true values, but they are not known by the solver until they have been solved, and as soon as their true value has been ascertained, it is printed in the middle of the blank square.

Prior to annotation, I lump together all the techniques for eliciting the true values of a number of blank squares under the single term ***trapping***. This I would broadly define as the limiting of the territory that a ***true value*** may occupy, decreasing that territory by ezcluding the rows, columns, and boxes occupied by other squares bearing the same ***true value***, until that territory has been reduced to a single square, which then betrays its ***pre-existing true value***, which is then inscribed in the center of that square.

During annotation, all the possible ***true values*** a blank square may possess are written as small digits at the top of it. These small digits are called ***candidates***. Later, after annotation, a myriad of other techniques are then employed to reduce the number of ***candidates*** at the its top of each square to one single candidate. That one remaining candidate is the ***true value* of the square**, and when it is discovered, it is removed from the top of the square and printed as a single large digit in the center of the square. Note that the ***blank squares*** are the only ones with candidates at their tops, and they must have them after the process of annotation has been completed.

During a ***temporary update***, ***temporary candidates*** may also exist in the centers of squares, ***candidates*** of the same size as those at the tops of squares. ***Candidates*** and ***temporary candidates*** often coexist during a temporary update. There may be blank squares without temporary candidates, but never without candidates at their tops. At the end of a temporary update, all temporary candidates are erased, prior to the next ***temporary update***. When a temporary update has proved its validity, by its adherence to 9-perfection, all the temporary candidates in the centers of squares replace the candidates at their tops. See the **Subpattern Analysis Textbook**, pages 22-24, for a discussion of temporary updates. (Throughout that textbook, the term ***established value*** is an equivalent term, a synonym for ***true value***.)

Solving a sudoku means discovering the ***true values*** of all 81 squares of the sudoku.. At the beginning, only the ***givens*** are known. These are the squares which, before the solver has done any solving, have their ***true values*** displayed in their centers, the clues to solving the sudoku. At the end of solving, every square will have, printed in the middle of it, its own ***true value***.

Even though the solver may not know the ***true value*** of a square, it is, in a shadowy way, predetimined by the puzzle master, the creator of the puzzle, and it is there from the very start, logically implied by the ***givens***, by both their positions and their own (as yet unknown) ***true values***.

The ***neighborhood*** of a specific square is the collection of all other squares in the row, column, and box of that specific square which possess the same true value. Just as there are 81 different squares, there are 81 different ***neighborhoods***, one for each square. These ***neighborhoods*** can be quite overlapping at times, and are never entirely disjoint from one another, but they only overlap rows with columns, or columns with rows, or share rows or columns in common, since boxes themselves never overlap.

It is useful to glance over Chapter 1 of the **Subpattern Analysis Textbook**, as well as the section within Chapter 2 dealing with temporary updates. It is also useful to glance over the extremely detailed examples of temporary updates offered in **Fully Solved Subpattern Analysis**. No discussion of temporary updating is offered herein, although its purpose and usage should be reasonably clear from the many examples. Analysis by subpattern, and analysis by separation are, at least abstractly, very alike, and all the updates in this text are made in the language of temporary updates.

Trapping is the process of limiting the known territory that squares possessing a particular digit as their true value may occupy, systematically decreasing that territory by excluding all rows, columns and boxes already occcupied by other squares in their neighborhood possessing that same digit as their true value, until that territory has been reduced to a single square, which may then be deduced as possessing that same true value, like trapping a desired zoo specimen by gradually reducing the territory it is free to roam in, often by the use of beaters. Trapping consists of the collection of all rules and techniques developed for the reduction of unsolved squares prior to annotation.

It is true that trapping could be dispensed with entirely, by beginning with annotation alone. Many squares with unique values would be discovered by virtue of their having only one candidate, and others would be discovered by the number of their candidates being reduced to one.. It would, however, be a horrid experience, in that it would make the starting number of candidates overwhelming. It would also rob the solver of the joy of trapping, one of the more pleasurable aspects of solving, because it is geometric rather than algebraic, visual rather than cerebral. It should also be pointed out that trapping often continues long after annotation, usually in a reduced, yet still useful role.

Logic terminology.

A ***9-string*** is a general term representing any row, column, or box (by virtue of its possessing 9 squares). It is never used for any set of nine squares which is not a row, column, or box.

A ***9-string*** is said to be ***9-perfect*** if it possesses every digit from 1 to 9.

A sudoku is ***9-perfect*** if every one of its 9 rows, 9 columns, and 9 boxes is ***9-perfect.***

9-perfection is the quality of being ***9-perfect***. A 9 x 9 grid composed of the digits 1-9 is not a sudoku unless it is ***9-perfect***.

9-imperfection is the state of a ***9-string*** or an entire sudoku not being ***9-perfect***.

The solution to a sudoku is required to be ***9-perfect***.

A ***contradiction*** describes as invalid any action or assumption which does not result in singularity of solution**.** This stems from the fundamental agreement by the sudoku community that an unsolved sudoku is guaranteed to have one and only one solution.. It is this agreed-upon assumption that allows the discovery of a fatal four to be considered a logical contradiction, requiring the rejection of any assumption which leads to it.

A ***contradiction*** also describes any result which is not ***9-perfect***.

An action or situation is considered ***invalid*** or ***logically false*** when it leads either to 9-imperfection, or to multiple solutions.

Comparisons between solving by separations and solving by subpatterns.

As I developed the method of solving by separations, and compared it to solving by subpatterns, I saw many differences between the two. The most obvious is the variety in the number of possible subpatterns, depending on the pattern and its complexity, where the number of columns in a separation is always limited by 9 – the number of squares is a row, column or box.

The similarities are at the logical level. The common requirement is that every possible distribution be found, so that it may be tested, just as every possible subpattern must be discovered, so that it may be tested. The remaining one to be tested, provided no errors were made, is known in advance to be be the solution to the sudoku, even before it is tested, but one must still go through the motions of testing, because the result is the solution to the sudoku.

In the testing of both, the failures at first zip through the puzzle, looking like real winners, and then fizzle out at the final moment/ In the testing of both, it is often the winning separation or subpattern which seems to resist at every step, the solving running out of steam quickly, and then, when all seems lost, an obscure distribution of candidate squares grudgingly offers up a small success and then another leading to the elusive solution. The net must be fine enough to catch the fish.

Testing by candidate separation is that of separating the members of the individual groups in such a way that one ends up with a single digit in each group, and the group assumes its identity as a solved square. But that single digit has to be an original member of the group it comes to represent. Groups are only whittled down, not added to from the outside. Identical digits in different groups are cleverly assassinated in such a manner as to maintain one representative in each square, always the same number of squares as there are digits.

In both separation testing and subpattern testing, one first untangles and then tests. Only one can be correct. The test of a particular subpattern is whether it causes to emerge, through its interaction with the other members of the sudoku, a single member for every single square in the entire sudoku. The test of a particular separation is whether it is in complete harmony with the already existing members of the sudoku. In both cases it is a survival of the fittest.

In a strange way, it seems as if the method of selection of the winning separation is more than just a another gimmick. It has an extremely close relationship with the method of selection of the winning subpattern. One is a twisted metaphor of the other..Separations are really arrangements by position. Subpatterns are arrangements by digit. The arrangements are by row and column. Boxes have a higher level arrangement of position, but are entirely similar to rows and columns with respect to separations, as well as entirely similar to them with respect to subpatterns. There is a principle of duality in the sudoku regarding the three kinds of 9-strings – rows, columns and boxes. One can almost take any true statement involving them, then rearrange their order in the statement, and come out with an another true statement. This is only true for certain rearrangements, of course. Still, there seems to be a secret relationship among them – the rows, columns, boxes, and subpatterns.

I've been solving all my favorite daily puzzles for the last month using the separation method every time I would have ordinarily used a solution by subpattern, and have found it to be easier and faster..