Experiments on the Generation of Distinguishing N-Tuples for Selected Character Dichotomies

Technical Report No. ECSE-OCR-18DEC95

Andrew Shapira
ECSE Department
Rensselaer Polytechnic Institute
Troy, NY 12180

December 22, 1995

1 Introduction

This report is a companion to [1]. In the present document, we describe experiments on generating n-tuples for optical character recognition. The focus is on the generation process itself, and not the use of the generated tuples. The experiments were conducted at Rensselaer during late January, 1995.

An outline of the report is as follows. In Section 2 we describe the experiments. The presentation of the experiment data is explained in Section 3. Section 4 contains some discussion. The experiment data is presented in Appendix A.

This report assumes familiarity with [1]. Additional information regarding the generators, and further discussion of the experiments, may be found there.

2 Description of Experiments

We consider the generation of n-tuples using two generators, called Gen0 and Gen1. Gen0 is a simple backtracking algorithm. Gen1 is a more sophisticated backtracking algorithm.

The main purpose of the experiments was to show the following.

1. Distinguishing tuples can be found reasonably quickly, when they exist.
2. Gen1 is a reasonably efficient algorithm for generating tuples. This was to be shown by comparing Gen1 to the benchmark program Gen0.
3. The execution time of Gen1 can be controlled by varying the difficulty of the problem as defined by the p and q parameters.

We also consider the quality of solution tuples as a function of the dichotomy.

The experiments consisted of executing Gen0 and Gen1 on selected character dichotomies for $n = 4, 7, 10$. Five dichotomies were used, giving $2 \cdot 3 \cdot 5 = 30$ experiments. For each experiment, Appendix A contains one table summarizing the results. Combined, the experiments represent more than 40,000 generator invocations, and hundreds of hours of CPU time. (Unless otherwise specified, all times in ths document are CPU times.)

The experiments were conducted on 4 two-processor SPARC 20's, using one CPU at a time. Three of the SPARC 20's had 64 megabytes of memory, and one had 256 megabytes. The generation process had modest memory requirements, e.g., a megabyte or less, so the amount of memory possessed by the machines was not critical for these experiments.

The generators were executed in the mode "find a solution for a specified (p, q) pair." The generators executed until either a solution was found, or the search terminated because a maximum search node limit was reached. The maximum node limits were selected so that the generators executed for about 5 minutes on a certain benchmark problem, when no solution was found. This “failure time” is different for different problems, because the time spent at a given search node varies. For these experiments, failure times were usually between 2 and 25 minutes. In some cases, when the problem was extremely constrained as when q is very small, failure times were only a few seconds.

For a given dichotomy, value of n, and generator, the generator was executed according to the following C-like pseudocode.

```
integer p,q,s,f;   // for given p,q: s=successes, f=failures

for (q = n-1, p = 0, s = 50; (q >= 2) and (p != 1); q = q-1) {
  for (p = 1; s == 50; p = p+1+4*(p>=10)) {
    for (s = f = 0; (s < 50) and (f < 51); ) {
      invoke generator for (p,q);
      if (generator found tuple)  s = s+1;
      else                        f = f+1;
    }
    report average times for this (p,q);
  }
}
```

We can think of the experiment as moving in a table where the rows correspond to values of p and the columns correspond to values of q. (This table underlies the reporting of the data in Appendix A.) We start at $q = n - 1$, and move down in the table (fixed q, increasing p). If, for a given p, q pair, we get 50 successes with a 50% failure rate or less, then we move down in the current column (we increase p). When a greater than 50% failure rate occurs, the current column is completed; we move left (decrease q) and start at the top of the new column (set p to 1). The table is completed when we finish the $q = 2$ column, or stop at row 1 in some column.

The five dichotomies used for the experiments are listed below.

1. c-e. 2. e-c. 3. e_5-c_5. 4. *acenou-sxz*. 5. c-n.

These dichotomies were chosen to represent a spectrum of problem types. Specifically, they were chosen for the following reasons. The c-e dichotomy is believed to be a difficult one to find tuples for. The dual of this dichotomy, e-c, was included because it seemed useful to see how the experiment results change when the positive and negative classes are switched. The experimental results show that finding tuples for a given dichotomy may be much easier or harder than for the dichotomy's dual, at least with the generators used here. The e_5-c_5 dichotomy is different from the other four dichotomies because it consists of fifth generation photocopies. The *acenou-sxz* dichotomy has several characters in each class. The c-n dichotomy is relatively easy to find tuples for.

The experiments used 8-point Times Roman characters scanned at 300 dots per inch. The characters are shown in Figure 1. The fifth generation photocopies c_5 and e_5 are shown in the bottom row of the figure. All characters were trimmed to the smallest rectangular bounding frame before being presented to the generators.

For a given dichotomy, the generators drew tuples from a rectangular region of pixels Π. The height (width) of Π is the smallest character height (width), taken over the positive exemplars of the dichotomy. For the purpose of determining the p and q values of a given tuple relative to a given character c, the character is considered to be embedded in an infinite sea of white pixels. Effectively, the tuple is tested in all shift offsets that place some part of Π over some part of c's smallest (inclusive) bounding box. Rotations are

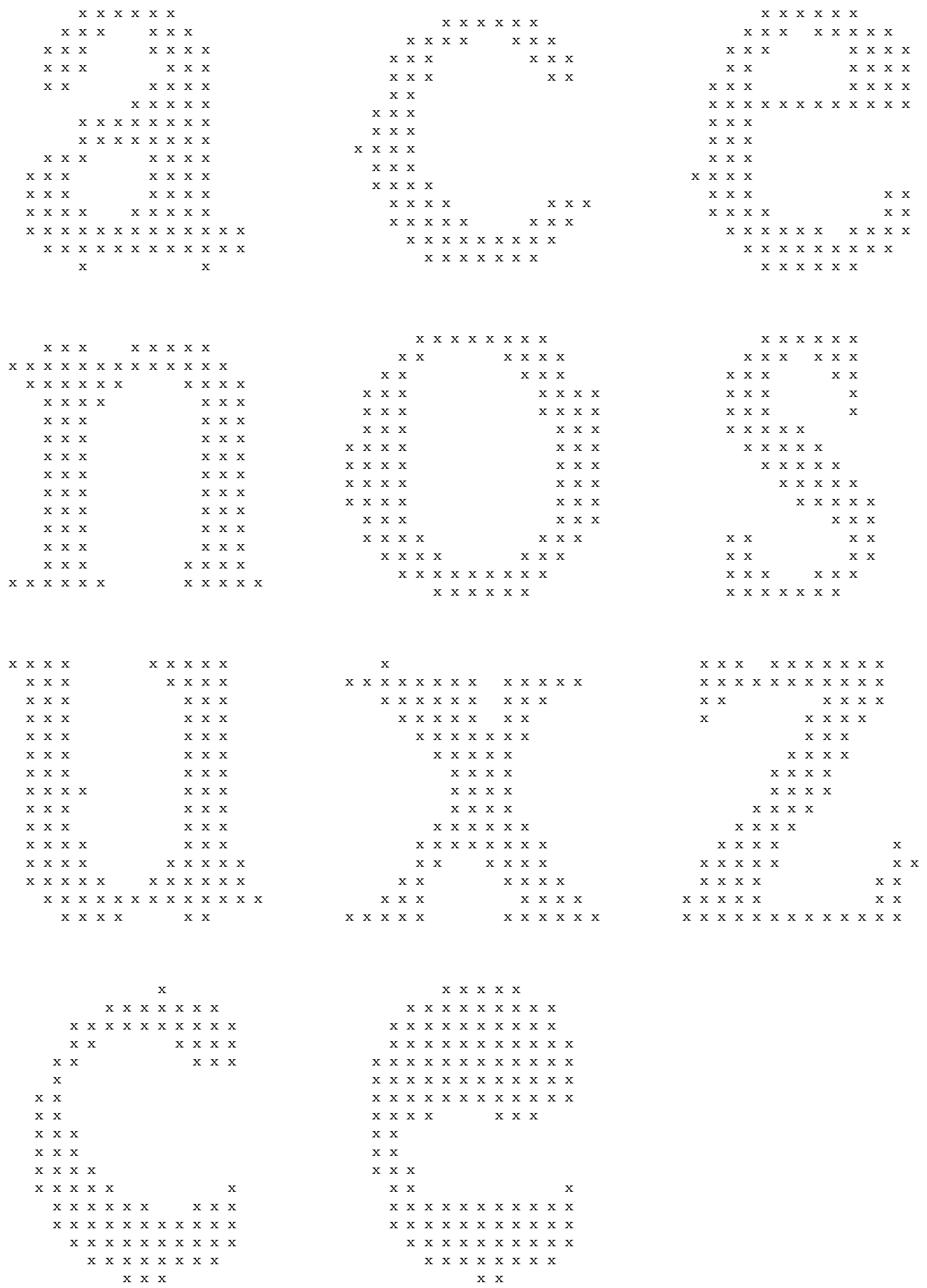

Figure 1: Characters used in experiments.

not considered, i.e., the tuple shifts are aligned with the two coordinate axes of the character.

The maximum node limit for Gen0 was 1,500,000; for Gen1, 70,000. The timeslice width for Gen1 was 500, and the width for restricted backtracking was 20. The parameters 500 and 20 were selected because of an empirically obtained belief that they give "reasonable" performance for many problems.

We did not know in advance how the generators would perform with the selected parameters on any particular problem in the experiments. The parameters were selected without regard for the particular problems included in the experiments.

In addition to the UNIX command-line arguments for the search limits given above, the following arguments were also used to invoke the generators.

```
-postkind pi -negtkind pi -randseed -one -bpnone -bw
```

The `postkind pi` and `negtkind pi` arguments specify that Π and the tuple shifts are as described earlier. The `randseed` argument means that each time the generator is invoked, it uses a new seed for producing pseudorandom numbers. Thus, on successive executions, the generator may follow different execution paths. Executing the generator many times gives a sample of the generator's possible behavior over the possible random number sequences. The `bw` argument indicates the use of black and white tuples (as opposed to all-black or all-white tuples). The `one` and `bpnone` arguments are instructions related to backtracking; their precise meaning is unimportant here.

3 Description of Experiment Data

The data is presented in Appendix A. There is one table for each (dichotomy, n, generator) triple. All times are in CPU seconds.

It is convenient to explain the data by example, so we refer to Table 1. At the top of the table is the mean execution time for the $p = 1, q = n - 1$ case; here, .2132 CPU seconds. The $p = 1, q = n - 1$ case is typically the easiest (p, q) pair to find tuples for. The times within the table are scaled to this case, to clarify the changing difficulty of finding tuples as we move away from $p = 1, q = n - 1$.

Within the table, the nonblank table entries can be categorized into three types (not including shading). The first type of entry has two numbers stacked above each other, as with $p = 1, q = 2$. The $p = 1, q = 2$ entry indicates that no failures occurred for $p = 1, q = 2$; the average scaled time of the 50 successes was 103 (103 times the base time of .2132 seconds), and the standard deviation of the scaled time was 73. Another type of table entry has five numbers. For example, with $p = 10, q = 3$, the mean scaled time for the 50 successes was 544, with a standard deviation of 395. There were 42 failures; the average scaled time for these 42 failures was 1313; and the standard deviation was 37. A third type of table entry occurs when there are no successes, as with $p = 2, q = 2$. For $p = 2, q = 2$ there were 51 failures; the average failure time was 1074, with a standard deviation of 24.

When available, the tables show the optimal (largest) p values for a given value of q. Optimal values are indicated by dark shading in the appropriate table entry. (No trials were conducted for certain optimal (p, q) pairs; this happens when the generators did not achieve optimality, or when the optimal p value is larger than 10 and is not a multiple of 5.) When it is known that no solution exists for a given value of q, this is indicated by an asterisk next to the column header. The optimality of the displayed values was determined by exhaustive search. In some cases, when it was not possible to determine the optimal values, the best known solution values are indicated with light shading. For example, see Table 2.

Considerable CPU time was used in the attempt to determine optimal values. In some cases, weeks of CPU time were spent determining whether a solution exists for a given (p, q) pair.

The procedure for determining optimal values was independent of the C-like pseudocode in Section 2. This procedure followed the staircase pattern described in [1]. For example, in Table 1, to find the optimal values, the generator traversed (p, q) pairs in the following order: $(1, 2)$, $(2, 2)$, $(2, 3)$, $(3, 3)$, $(4, 3)$, $(5, 3)$, $\cdots$, $(11, 3)$, $(12, 3)$. The optimal pairs are therefore $(1, 2)$ and $(11, 3)$, since no solutions were found for $(2, 2)$ or $(12, 3)$.

The tables were produced by a *perl* sript that scans the 50 to 101 generator output files for each relevant (p, q) pair in a given table, and automatically produces the LaTeX source to make the table.

4 Discussion

In this section we look at the three questions that the experiments were designed to answer. In Section 4.4, we consider the quality of solution tuples as a function of the dichotomy.

4.1 Tradeoff Between Difficulty and Execution Time

Apparently, solution tuples are relatively abundant when p, q is close to $p = 1, q = n - 1$. As we move away from this point, ostensibly the problem becomes increasingly constrained and the solution tuple density decreases. For a given dichotomy and value of n, there is a tradeoff between execution time and the difficulty of the problem as defined by the (supposed) solution density. Inspection of the tables shows that Gen0 and Gen1 require more execution time as we move away from $p = 1, q = n-1$. Thus, we can control the amount of time to find tuples by selecting p and q.

4.2 Time to Find Tuples

Here we address the issue of whether or not there exists an algorithm that finds distinguishing tuples reasonably quickly, when such tuples exist for a given dichotomy and value of n. Inspection of the tables in Appendix A yields an affirmative answer, at least for the problems in these experiments.

When solutions are relatively abundant or when solutions exist in a constrained situation such as for small n, either Gen0 or Gen1 is sufficient to find tuples. For the other cases, Gen0 does not do as well as Gen1.

4.3 Gen1 vs. Gen0

The third purpose of the experiments was to show that Gen1 is a "good" algorithm by comparing it to the benchmark algorithm Gen0.

For the most part, the measure of an algorithm should be how it does on difficult problems. (It is easy to do well on easy problems.) The definition of a difficult problem depends in general on the algorithm used to solve it. Here, we say that the difficulty increases when n increases and when (p, q) moves away from $p = 1, q = n - 1$. One reason for this is that (empirically) these cases take more time; in every table in Appendix A, as we move away from $p = 1, q = n - 1$ we see increased time to find a solution. Another reason is that the tuples away from $p = 1, q = n - 1$ are the ones that are most desirable for OCR [1].

Examination of the tables shows that for the difficult problems as defined above, Gen1 finds solutions faster and over a wider range than Gen0. For the "easy" problems, Gen0 does better.

It is instructive to examine why Gen0 does better than Gen1 on easy problems. One reason is that Gen1 was designed with difficult problems in mind. A consequence of this design is that Gen1 cannot find any solution without evaluating many tuples at each search node, even when the search goes directly to a solution with no backtracking. On the other hand, Gen0 only evaluates one tuple at each search node. These facts and the experimental data suggest that we might want to form a hybrid algorithm that executes Gen0 and Gen1 in parallel. If t_0 and t_1 are the respective execution times for Gen0 and Gen1 on a given problem, then the hybrid algorithm takes time that is at worst roughly $2\min(t_0, t_1)$. This is in some ways more desirable than the execution characteristics of either generator. For easy instances, the hybrid uses at most twice the time of Gen0; for difficult instances, the hybrid uses at most twice the time of Gen1.

There are easy instances in these experiments where Gen0 finds tuples over a wider range than Gen1 does, e.g., the c-e dichotomy for $n = 4$. For these instances, the failure times for Gen0 are on the order of 15 or 20 minutes, whereas the failure times for Gen1 are on the order of a few seconds. This suggests that for these instances the search width parameter of 20 used for Gen1 is too restrictive; this value causes Gen1's search tree to be small and to contain no solutions. We suspect that Gen1 would find tuples for these instances if the search width were increased. The experiments appear, in these instances, to be unfair to Gen1, since Gen0 is allowed to spend more time searching.

4.4 Quality of Tuples as a Function of the Dichotomy

Here we briefly consider how, if at all, the quality of solution tuples depends on the dichotomy.

The best known values of p for a given value of q are summarized in Figure 2 ($n = 4$), Figure 3 ($n = 7$), and Figure 4 ($n = 10$), for each of the five dichotomies. The figures represent the shaded boxes that appear in the tables in Appendix A. Figure 2 ($n = 4$) represents optimal values; Figures 3 and 4 contain some entries that may not be optimal.

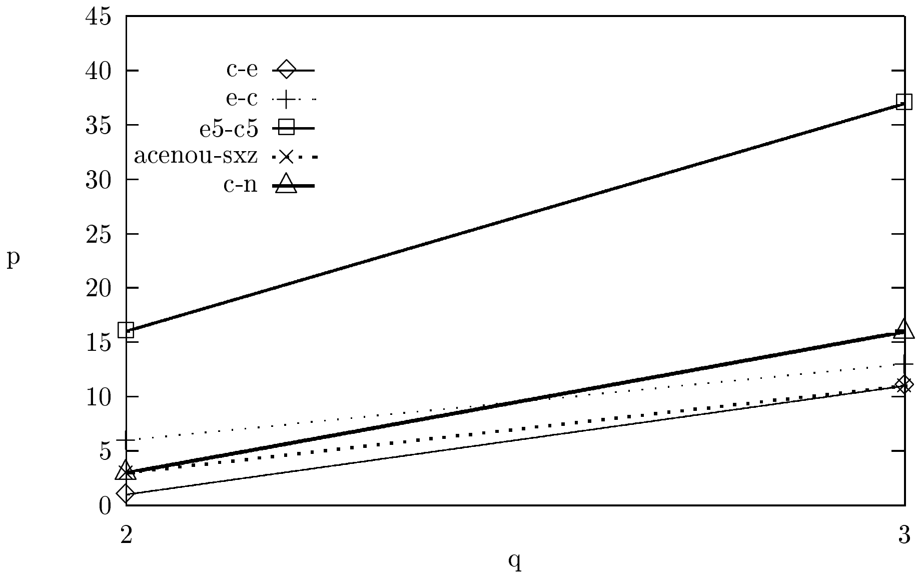

Figure 2: Optimal p for given q $(n = 4)$.

A criterion for comparing the quality of tuples available to two dichotomies A and B is as follows: we can consider A to have higher quality tuples available than B if, in a plot of optimal values such as Figure 2, the curve for A is everywhere at or above the curve for B.

According to this criterion, the e_5-c_5 dichotomy has better tuples available than the other four dichotomies. Using Figures 2, 3, and 4, the five dichotomies can be roughly ordered in increasing order of quality of available tuples as follows:

$$c\text{-}e \;<\; acenou\text{-}sxz \;<\; e\text{-}c \;\leq\; c\text{-}n \;<\; e_5\text{-}c_5.$$

This ordering is consistent with our earlier belief that the c-e dichotomy is a difficult one to find tuples for. The plots suggest that higher quality tuples are available for the e-c dichotomy than for its converse (except when $q = n - 1$).

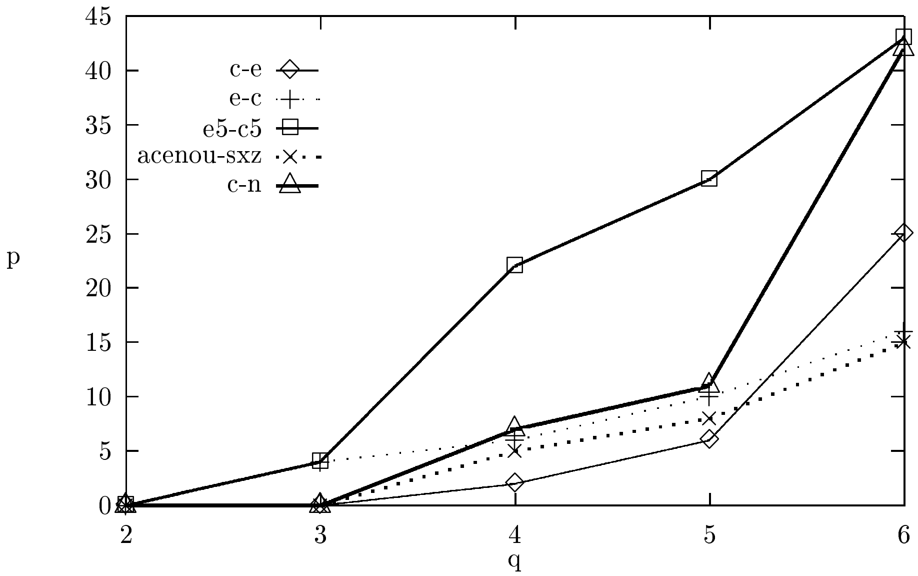

Figure 3: Largest known p for given q ($n = 7$).

The change in behavior at $q = n - 1$ is unexplained. It may be an artifact of the fact that the plots use only the best known values of p, and not necessarily the optimal values. This is another reminder that the data reported in this section is not exact and should be treated cautiously.

Despite their non-exact nature, the plots make it fairly clear that the quality of available tuples varies widely across dichotomies.

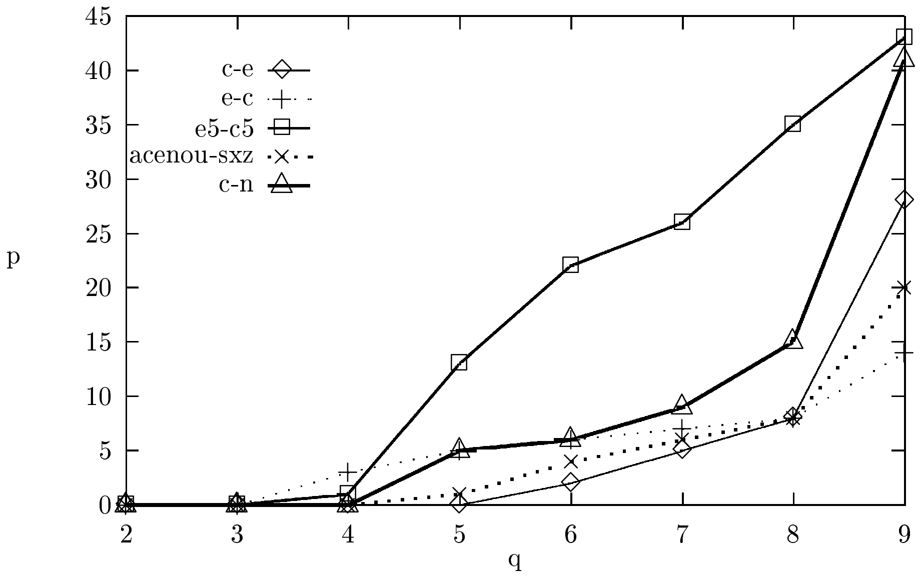

Figure 4: Largest known p for given q ($n = 10$).

5 Acknowledgement

I would like to thank D. Jung, M. Krishnamoorthy, and G. Nagy for commenting on a preliminary version of this report. G. Nagy participated in the design of the experiments, and suggested the plots in Section 4.4. Scanned characters were supplied by D. Jung.

References

[1] D. Jung and M. Krishnamoorthy and G. Nagy and A. Shapira, "N-Tuple Features for OCR Revisited," Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence, 1995.

Appendix A: Experiment Data

This appendix contains 30 tables, one for each (dichotomy, n, generator) triple.

Note: to reduce the table width, Table 21 does not have a column for $q = 3$. The data that would go in this column is that for $p = 1, q = 3$ there were 51 failures with an average scaled time of 140 and a standard deviation of 2.5. Exhaustive search determined that there are no solutions for $q = 3$.

Mean time for p=1,q=3: .2132 s

	q	
	2	3
1	103 73	1 0.08
2	1074^{51} 24	1.0 0.17
3		1.6 1.2
4		2.6 3.2
5		4.1 3.8
6		7.6 7.8
7		23 22
8		149 230
9		182 255
10		544 1313^{42} 395 37
11		
15		1296^{51} 40
20		
25		
30		
35		
40		

p (row label)

Table 1: Success and failure times, n = 4, c-e, Gen0.

Mean time for p=1,q=6: .169 s

p \ q	2*	3*	4	5	6
1		1390^{51} 89	327 1328^{33} 345 80	2.7 3.4	1 0.19
2			1338 1330^{51} 0 132	57 119	1.1 0.26
3				185 1495^{5} 246 206	1.3 0.77
4				375 1366^{26} 382 154	8.5 1634^{1} 30 0
5				467 1403^{51} 482 174	49 1469^{4} 135 142
6					43 1083^{4} 75 87
7					100 1153^{11} 223 143
8					172 1230^{9} 276 247
9					186 1211^{30} 281 251
10					298 1314^{48} 281 229
15					273 1246^{51} 152 198
20					
22					
25					
30					
35					
40					

Table 2: Success and failure times, n = 7, c-e, Gen0.

Mean time for p=1,q=9: .2066 s

	q					
p	4	5	6	7	8	9
1			442 1283^{51} 352 186	113 1354^{3} 261 106	2.2 4.4	1 0.097
2				307 1347^{48} 336 293	16 1049^{1} 51 0	1.0 0.09
3				448 1345^{51} 364 281	130 1509^{26} 214 334	3.7 1652^{4} 15 107
4					260 1296^{51} 343 306	43 1283^{5} 263 325
5						68 1838^{9} 229 149
6						125 1396^{16} 321 331
7						125 1388^{28} 312 324
8						163 1202^{32} 318 302
9						107 1375^{51} 190 350
10						
15						
20						
25						
28						
30						
35						
40						

Table 3: Success and failure times, n = 10, c-e, Gen0.

Mean time for p=1,q=3: .1712 s

	q	
	2	3
1	6.9 6.9	1 0.13
2	8.1 6.3	0.99 0.13
3	8.3 5.8	1.0 0.16
4	6.7 5.5	0.98 0.17
5	7.6 5.2	1.0 0.16
6	37 39	1.1 0.16
7	1190^{51} 38	1.1 0.22
8		1.5 1.0
p 9		1.9 1.4
10		4.7 4.2
13		
15		1362^{51} 52
20		
25		
30		
35		
40		

Table 4: Success and failure times, n = 4, e-c, Gen0.

Mean time for p=1,q=6: .2062 s

p	q = 2*		3		4		5		6	
1		1277^{51} 19	174 238	1278^{10} 22	2.6 2.2		1.0 0.13		1 0.033	
2			186 201	1228^{42} 30	18 28		1.9 2.8		4.3 23	
3			413 306	1176^{51} 73	60 108	1215^{1} 0	3.4 11		1.1 0.24	
4					192 294	1302^{4} 18	12 25	1690^{1} 0	6.3 23	
5					333 385	1242^{23} 123	23 48	1458^{1} 0	6.4 34	1124^{1} 0
6					418 411	1249^{51} 132	175 266	1057^{6} 119	27 123	
7							378 362	1273^{41} 184	38 106	1357^{3} 142
8							723 396	1313^{51} 190	42 93	1070^{6} 218
9									145 279	1173^{14} 243
10									212 329	1180^{20} 285
15										1205^{51} 277
16										
20										
25										
30										
35										
40										

Table 5: Success and failure times, n = 7, e-c, Gen0.

Mean time for p=1,q=9: .2076 s

	q					
p	4	5	6	7	8	9
1	290 1303^{51} 364 74	184 1242^{6} 225 122	11 24	13 83	1.0 0.16	1 0.04
2		397 1182^{51} 349 95	109 1250^{5} 225 205	23 1078^{2} 123 180	20 98	1.0 0.066
3			248 1232^{20} 223 177	79 1145^{7} 175 203	22 1548^{9} 82 246	3.9 15
4			415 1244^{51} 374 215	217 1378^{21} 309 260	35 1328^{14} 94 268	11 1070^{1} 33 0
5				306 1254^{51} 274 259	102 1207^{40} 207 300	14 1435^{7} 44 345
6					191 1276^{51} 230 276	82 1231^{22} 209 380
7						144 1232^{36} 263 379
8						198 1156^{51} 267 310
9						
10						
14						
15						
20						
25						
30						
35						
40						

Table 6: Success and failure times, n = 10, e-c, Gen0.

Mean time for p=1,q=3: .2198 s

	q = 2		q = 3	
p				
1	1.1 0.13		1 0.044	
2	1.2 0.23		0.98 0.083	
3	1.3 0.29		1.0 0.054	
4	1.5 0.48		1.0 0.06	
5	1.8 0.65		1.0 0.057	
6	2.1 1.3		1.0 0.073	
7	2.1 1.1		0.99 0.083	
8	2.2 1.3		1.0 0.12	
9	3.1 1.8		0.99 0.071	
10	3.2 1.9		1.0 0.08	
15	9.8 15		1.3 0.8	
16				
20		1140^{51} 8.9	1.2 0.6	
25			1.8 1.3	
30			4.2 3.1	
35			371 391	1263^{18} 30
37				
40				1223^{51} 48

Table 7: Success and failure times, n = 4, e_5-c_5, Gen0.

Mean time for p=1,q=6: .1924 s

p \ q	2*	3	4	5	6
1	1299^{51} 47	99 106	2.0 1.6	0.99 0.19	1 0.19
2		307 1156^{7} 300 76	4.9 11	1.0 0.2	0.99 0.18
3		511 1119^{51} 335 58	16 32	1.1 0.34	1.0 0.32
4			9.6 16	1.5 2.7	1.0 0.24
5			14 25	25 164	1.0 0.36
6			15 25	1.2 0.58	2.6 11
7			22 38	2.3 4.1	3.0 1268^{1} 9.3 0
8			34 64	35 124	5.3 24
9			59 83	22 74	1.6 1320^{1} 2.9 0
10			59 1071^{2} 109 93	3.7 910^{1} 5.4 0	5.3 1118^{2} 18 179
15			278 1075^{10} 285 150	55 913^{1} 171 0	65 1008^{4} 154 93
20			496 1047^{50} 328 141	100 1021^{9} 166 220	53 1038^{10} 136 178
22					
25			1070^{51} 154	354 1086^{51} 254 189	83 1072^{13} 202 194
30					161 1067^{51} 213 220
35					
40					
43					
45					

Table 8: Success and failure times, n = 7, e_5-c_5, Gen0.

Mean time for p=1,q=9: .2212 s

p \ q	4	5	6	7	8	9
1	500 1333^{51} 254 75	214 1313^{12} 326 123	28 131	24 139	1.0 0.063	1 0.065
2		314 1291^{23} 362 118	46 1447^{1} 146 0	22 116	2.9 10	1.0 0.086
3		410 1197^{38} 342 133	63 1197^{2} 159 157	8.4 1527^{2} 36 191	23 120	1.1 0.6
4		559 1170^{51} 448 101	155 1328^{3} 280 95	15 1719^{2} 65 63	13 1198^{4} 55 389	1.1 0.47
5			68 1151^{9} 147 199	26 1491^{1} 98 0	16 1485^{4} 91 245	1.3 1027^{4} 1.2 190
6			132 1136^{8} 208 236	28 1225^{5} 68 318	40 1194^{3} 151 201	1.7 1267^{3} 4.0 523
7			198 1119^{19} 254 156	33 1256^{7} 65 314	48 1287^{10} 150 253	5.6 1343^{3} 32 285
8			222 1102^{12} 240 174	36 1123^{11} 76 203	45 1080^{7} 147 242	2.3 999^{6} 5.2 235
9			168 1050^{42} 190 159	105 1115^{12} 271 261	53 1168^{8} 116 226	8.4 1264^{6} 30 449
10			279 1091^{31} 269 174	67 1082^{7} 128 256	50 1185^{17} 138 290	42 1376^{9} 140 348
13						
15			181 990^{51} 195 150	112 971^{51} 158 164	70 1244^{21} 166 343	18 1061^{18} 74 301
20					131 1154^{51} 245 302	78 1095^{47} 193 277
22						
25						59 1014^{51} 121 232
26						
30						
35						
40						
43						
45						

Table 9: Success and failure times, n = 10, e_5-c_5, Gen0.

Mean time for p=1,q=3: .7872 s

	q	
	2	3
1	6.2 5.7	1 0.28
2	11 12	1.3 0.67
3	23 23	1.4 0.72
4	564^{51} 7.4	1.8 1.1
5		3.4 3.7
6		7.5 15
7		12 31
8		21 44
9 p		58 110
10		137 835^{5} 185 26
11		
15		822^{51} 22
20		
25		
30		
35		
40		

Table 10: Success and failure times, n = 4, acenou-sxz, Gen0.

Mean time for p=1,q=6: 1.9652 s

	q				
p	2*	3*	4	5	6
1		324^{51} 42	136 345^{14} 121 27	9.7 272^{2} 23 14	1 3.0
2			131 348^{21} 121 27	33 249^{8} 51 31	2.3 6.3
3			138 338^{51} 114 34	64 279^{22} 80 70	6.3 236^{4} 15 57
4				89 282^{38} 98 68	18 236^{9} 46 53
5				114 313^{51} 115 69	25 276^{13} 48 53
6					35 252^{27} 53 47
7					51 263^{51} 72 50
8					
9					
10					
15					
20					
25					
30					
35					
40					

Table 11: Success and failure times, n = 7, acenou-sxz, Gen0.

Mean time for p=1,q=9: 2.1424 s

	q					
	4	5	6	7	8	9
1				43 216^{51} 58 40	13 193^{14} 34 36	1 169^{1} 2.6 0
2					39 217^{41} 52 42	6.6 219^{3} 18 68
3					57 203^{51} 67 45	14 186^{18} 33 50
4						18 180^{38} 37 46
5						30 177^{51} 40 43
6						
7						
p 8						
9						
10						
15						
20						
25						
30						
35						
40						

Table 12: Success and failure times, n = 10, acenou-sxz, Gen0.

Mean time for p=1,q=3: .1804 s

p	q = 2	q = 3
1	12 8.6	1 0.17
2	31 35	0.98 0.17
3	63 62	1.0 0.2
4	1297^{51} 36	0.94 0.16
5		1.0 0.2
6		1.1 0.5
7		1.1 0.58
8		1.4 1.2
9		1.5 1.4
10		2.4 2.2
15		154 221
16		
20		1292^{51} 90
25		
30		
35		
40		

Table 13: Success and failure times, n = 4, c-n, Gen0.

Mean time for p=1,q=6: .2064 s

p \ q	2^*		3^*		4		5		6	
1				1199^{51} 65	21 28		1.2 0.35		1 0.077	
2					93 131		1.6 1.5		1.0 0.12	
3					140 219	1141^{22} 110	3.4 5.8		1.0 0.091	
4					183 229	1200^{20} 111	45 122		1.0 0.082	
5					233 281	1168^{31} 109	58 132	1604^{1} 0	2.1 6.1	
6					402 370	1197^{51} 127	94 169	1101^{6} 148	1.3 1.0	
7							114 194	1073^{8} 192	7.5 34	
8							227 303	1127^{17} 234	8.2 30	
9							430 338	1241^{51} 196	22 54	1114^{2} 121
10									79 270	1368^{5} 284
11										
15									183 234	1372^{14} 315
20									332 325	1208^{51} 287
25										
30										
35										
40										
42										
45										

Table 14: Success and failure times, n = 7, c-n, Gen0.

Mean time for p=1,q=9: .1706 s

p \ q	4	5	6	7	8	9
1		510 1242^{51} 422 136	125 1066^{6} 239 131	7.2 953^{1} 17 0	1.0 0.28	1 0.16
2			170 1154^{26} 242 158	16 931^{1} 61 0	1.3 1.2	0.95 0.13
3			268 1147^{46} 295 143	106 1123^{11} 232 281	24 859^{1} 102 0	1.0 0.42
4			309 1186^{51} 315 178	161 1231^{24} 255 253	52 1037^{3} 113 81	1.1 0.4
5				155 1099^{51} 251 179	36 1049^{11} 85 154	1.7 1109^{1} 3.7 0
6					149 1177^{13} 249 282	4.1 954^{4} 17 101
7					184 1134^{32} 302 254	26 1119^{2} 117 233
8					128 1215^{51} 187 341	34 1571^{4} 118 487
9						24 1039^{7} 72 182
10						36 1199^{23} 82 293
15						142 1272^{29} 247 347
20						96 1297^{51} 168 363
25						
30						
35						
40						
41						
45						

Table 15: Success and failure times, n = 10, c-n, Gen0.

Mean time for p=1,q=3: .6368 s

	q	
	2	3
1	2.4^{51} 0.044	1 0.049
2		0.95 0.055
3		0.98 0.11
4		1.3 0.43
5		4.8^{51} 0.18
6		
7		
8		
9		
p 10		
11		
15		
20		
25		
30		
35		
40		

Table 16: Success and failure times, n = 4, c-e, Gen1.

Mean time for p=1,q=6: .5668 s

p \ q	2*	3*	4	5	6
1		15^{51} 1.8	86 64	1.1 0.2	1 0.096
2			166^{51} 7.9	7.8 3.8	0.99 0.12
3				224 352^{51} 6.3 16	2.5 1.8
4					2.3 1.4
5					1.5 0.73
6					2.3 0.62
7					3.1 1.3
8					3.5 1.9
9					3.8 2.5
10					5.4 4.1
15					94 467^{7} 66 27
20					457^{51} 17
22					
25					
30					
35					
40					

Table 17: Success and failure times, n = 7, c-e, Gen1.

Mean time for p=1,q=9: .8808 s

p \ q	4	5	6	7	8	9
1			83 191^{51} 3.0 7.1	0.98 0.04	1.0 0.041	1 0.054
2				33 221^{9} 23 16	5.4 2.6	0.92 0.078
3				171^{51} 13	5.3 0.59	2.2 1.4
4					6.9 3.0	2.2 1.4
5					8.1 158^{8} 9.0 9.6	1.1 0.47
6					150^{51} 8.9	1.9 0.42
7						2.4 0.97
8						3.0 1.3
9						2.2 1.1
10						2.9 1.4
15						3.9 3.3
20						5.5 125^{2} 4.0 3.5
25						109^{51} 10
28						
30						
35						
40						

Table 18: Success and failure times, n = 10, c-e, Gen1.

Mean time for p=1,q=3: .472 s

	q	
	2	3
1	1.2 0.14	1 0.12
2	1.2 0.15	1.1 0.12
3	1.2 0.14	1.0 0.14
4	1.2 0.12	1.0 0.11
5	3.0^{51} 0.29	1.0 0.1
6		1.1 0.13
7		1.2 0.12
8		1.2 0.13
9 p		1.2 0.18
10		1.2 0.17
13		
15		4.6^{51} 0.44
20		
25		
30		
35		
40		

Table 19: Success and failure times, n = 4, e-c, Gen1.

Mean time for p=1,q=6: .6548 s

	q				
p	2*	3	4	5	6
1	4.8^{51} 0.15	1.4 0.11	1.6 0.21	1.7 0.25	1 0.2
2		1.5 0.18	1.3 0.18	1.5 0.25	1.2 0.19
3		44 7.2	1.5 0.29	1.6 0.27	1.0 0.18
4		44^{51} 1.3	1.3 0.22	1.4 0.27	1.0 0.16
5			3.0 3.3	1.4 0.26	0.97 0.13
6			266 44	2.6 1.7	1.2 0.15
7			400^{51} 2.8	6.0 3.3	1.2 0.021
8				9.5 7.3	1.2 0.03
9				57 336^{51} 5.7 7.7	1.1 0.079
10					1.0 0.029
15					203^{51} 7.2
16					
20					
25					
30					
35					
40					

Table 20: Success and failure times, n = 7, e-c, Gen1.

Mean time for p=1,q=9: .7346 s

	q 4	5	6	7	8	9
1	4.3 2.9	1.7 0.23	1.8 0.21	1.9 0.18	1.9 0.21	1 0.29
2	87 151^{47} 43 7.1	1.4 0.22	1.4 0.18	1.6 0.24	1.5 0.21	1.2 0.27
3	267^{51} 9.8	1.9 0.72	1.7 0.23	1.7 0.24	1.6 0.26	0.96 0.23
4		42 334^{4} 67 9.2	1.4 0.21	1.4 0.2	1.4 0.18	1.1 0.21
5		378^{51} 19	2.9 3.3	3.1 3.6	1.3 0.18	0.98 0.17
6			415^{51} 20	50 416^{51} 18 19	2.0 1.7	1.4 0.036
7					4.9 3.6	1.3 0.017
8					8.1 3.3	1.2 0.079
9 *p*					318^{51} 12	1.1 0.13
10						1.0 0.052
14						
15						198^{51} 8.7
20						
25						
30						
35						
40						

Table 21: Success and failure times, n = 10, e-c, Gen1 (see note at beginning of Appendix A).

Mean time for p=1,q=3: .6522 s

p	q = 2	q = 3
1	1.2 0.087	1 0.1
2	1.1 0.093	1.0 0.12
3	1.1 0.081	1.0 0.095
4	1.0 0.053	1.1 0.1
5	1.1 0.045	0.91 0.037
6	1.0 0.056	1.2 0.02
7	1.1 0.11	1.2 0.024
8	1.1 0.073	1.2 0.028
9	1.2 0.14	1.2 0.021
10	1.3 0.18	1.2 0.021
15	1.4 0.1	1.1 0.021
16		
20	3.1^{51} 0.038	1.0 0.023
25		1.2 0.18
30		1.3 0.2
35		3.3^{51} 0.097
37		
40		

Table 22: Success and failure times, n = 4, e_5-c_5, Gen1.

Mean time for p=1,q=6: .6554 s

	q				
p	2*	3	4	5	6
1	2.8^{51} 0.51	8.9 6.1	2.8 0.95	1.0 0.25	1 0.25
2		8.5 3.6	2.0 1.1	2.3 1.1	0.98 0.27
3		54 20	1.5 0.7	2.0 0.89	0.99 0.25
4		156^{51} 11	2.2 0.88	1.3 0.26	1.2 0.34
5			3.6 2.5	1.6 0.33	0.78 0.14
6			1.4 0.76	1.3 0.23	1.3 0.23
7			3.3 1.2	1.2 0.21	1.2 0.21
8			2.5 2.0	1.2 0.21	1.2 0.22
9			2.4 1.9	1.1 0.21	1.2 0.21
10			3.6 2.5	1.1 0.15	1.2 0.21
15			5.3 4.4	1.9 1.2	1.1 0.19
20			5.4 1.2	2.3 1.5	0.99 0.17
22					
25			56^{51} 8.5	6.3 8.9	1.7 0.97
30				145^{51} 8.4	3.4 123^{1} 5.0 0
35					150^{51} 8.4
40					
43					
45					

Table 23: Success and failure times, n = 7, e_5-c_5, Gen1.

Mean time for p=1,q=9: .8522 s

	q					
p	4	5	6	7	8	9
1	30 132^{51} 18 2.4	2.6 0.99	2.6 0.93	2.7 0.85	1.1 0.27	1 0.24
2		2.0 1.2	2.4 1.2	2.3 1.2	2.2 1.2	1.0 0.27
3		3.7 2.8	1.7 0.75	2.0 0.94	2.0 0.99	0.94 0.19
4		6.8 3.0	2.2 0.87	2.3 0.71	1.5 0.092	1.4 0.3
5		11 5.3	3.5 2.2	3.2 1.7	1.9 0.2	0.77 0.022
6		10 4.1	1.7 0.83	1.7 0.85	1.5 0.021	1.5 0.049
7		13 7.5	3.8 1.3	3.7 1.2	1.4 0.062	1.4 0.064
8		28 31	3.5 2.4	2.7 1.7	1.4 0.064	1.4 0.05
9		63 239^{9} 49 13	3.2 2.5	2.8 1.9	1.3 0.03	1.4 0.041
10		42 262^{44} 19 20	4.6 3.0	4.1 2.7	1.3 0.047	1.4 0.055
13						
15		265^{51} 11	13 5.6	3.5 2.6	1.5 0.84	1.1 0.055
20			74 216^{51} 65 12	11 8.2	2.8 1.4	0.96 0.033
22						
25				130^{51} 15	2.6 132^{33} 1.1 16	1.7 0.81
26						
30					118^{51} 5.5	3.0 126^{7} 3.5 6.0
35						144^{51} 6.8
40						
43						
45						

Table 24: Success and failure times, n = 10, e_5-c_5, Gen1.

Mean time for p=1,q=3: 1.6832 s

		q	
		2	3
p	1	1.0 0.18	1 0.15
	2	1.1 0.2	0.93 0.093
	3	1.1 0.24	0.89 0.099
	4	2.5^{51} 0.4	0.89 0.12
	5		0.95 0.14
	6		0.91 0.15
	7		0.96 0.15
	8		0.9 0.15
	9		0.9 0.16
	10		0.96 0.18
	11		
	15		5.4^{51} 0.75
	20		
	25		
	30		
	35		
	40		

Table 25: Success and failure times, n = 4, acenou-sxz, Gen1.

Mean time for p=1,q=6: 2.235 s

p \ q	2*	3*	4	5	6
1		64^{51} 3.5	1.2 0.32	1.0 0.012	1 0.032
2			2.5 1.5	1.4 0.8	0.95 0.019
3			6.5 5.2	0.99 0.26	0.91 0.011
4			13 5.4	1.4 0.59	0.88 0.0089
5			12 3.0	2.0 1.3	1.8 1.2
6			110^{51} 1.5	5.1 6.5	0.87 0.028
7				135^{51} 2.9	0.87 0.0054
8					0.83 0.01
9					0.81 0.008
10					0.97 0.3
15					422^{51} 6.1
20					
25					
30					
35					
40					

Table 26: Success and failure times, n = 7, acenou-sxz, Gen1.

Mean time for p=1,q=9: 2.216 s

p \ q	4	5	6	7	8	9
1		196^{51} 9.5	70 32	3.0 2.1	1.0 0.16	1 0.17
2			47 39; 182^{5} 2.9	3.0 3.6	1.3 0.77	0.88 0.13
3			87 45; 173^{12} 11	14 9.0	1.1 1.0	0.89 0.14
4			186^{51} 19	9.3 7.3	1.7 0.56	0.84 0.14
5				13 3.6	1.8 1.1	1.5 1.2
6				139^{51} 6.1	6.0 7.9; 135^{2} 4.0	0.82 0.14
7					113^{51} 12	0.84 0.13
8						0.79 0.13
9						0.75 0.13
10						0.88 0.3
15						264^{51} 17
20						
25						
30						
35						
40						

Table 27: Success and failure times, n = 10, acenou-sxz, Gen1.

Mean time for p=1,q=3: .5378 s

	q	
	2	3
1	1.9 0.38	1 0.078
2	2.8^{51} 0.44	0.98 0.084
3		0.98 0.092
4		0.98 0.086
5		0.95 0.093
6		0.95 0.089
7		0.95 0.13
8		1.1 0.16
9 *p*		1.0 0.18
10		1.0 0.18
15		0.99 0.21
16		
20		6.4^{51} 1.1
25		
30		
35		
40		

Table 28: Success and failure times, n = 4, c-n, Gen1.

Mean time for p=1,q=6: .8056 s

	q				
	2*	3*	4	5	6
1		75^{51} 3.0	1.8 1.4	1.0 0.024	1 0.042
2			3.9 3.2	1.0 0.019	0.99 0.04
3			14 4.1	0.95 0.016	0.95 0.035
4			23 12	4.4 4.3	0.95 0.044
5			47 55	13 6.4	0.97 0.024
6			351^{51} 2.7	15 6.7	0.93 0.034
7				11 5.4	0.93 0.034
8				16 4.7	0.91 0.02
9				16 330^{14} 6.3 4.4	0.92 0.018
10 p				312^{51} 6.5	0.89 0.021
11					
15					2.7 1.8
20					13 6.3
25					100 92
30					429 535^{13} 99 11
35					232 414^{51} 161 13
40					
42					
45					

Table 29: Success and failure times, n = 7, c-n, Gen1.

Mean time for p=1,q=9: .8262 s

p	q = 4	5	6	7	8	9
1	173^{51} 26	70 144^{28} 49 11	1.9 1.4	1.1 0.63	0.97 0.17	1 0.065
2		182^{51} 24	10 3.7	1.1 0.54	1.0 0.18	0.94 0.046
3			16 3.7	5.5 4.3	0.99 0.17	1.0 0.14
4			62 266^{13} 67 14	8.1 5.0	1.2 0.48	1.1 0.12
5			104 328^{4} 110 13	16 6.6	13 8.4	1.0 0.19
6			335^{51} 13	19 4.3	17 7.0	0.9 0.16
7				357^{51} 10	14 6.2	0.92 0.17
8					17 6.4	0.88 0.16
9					18 365^{22} 7.2 13	0.94 0.17
10					342^{51} 16	0.91 0.16
15						2.2 1.6
20						23 14
25						91 494^{26} 60 20
30						346 435^{51} 91 27
35						
40						
41						
45						

Table 30: Success and failure times, n = 10, c-n, Gen1.