APPENDIX A

# Implementation Issues for Reverse Hillclimbing

Hopefully this appendix will be helpful to anyone who implements the reverse hillclimbing algorithm. The idea is conceptually simple, but relatively efficient implementation is a little tricky.

I have implemented the reverse hillclimbing in three different ways. The first, which was the method used to produce the results in [110], used a hash table in the manner described in §4.4.2(93). Using that approach, it is necessary to keep every vertex in the basin of attraction in memory until the recursion is complete. This is due to the possibility of vertices being revisited. This algorithm was controlled using the recursive strategy described for the basic algorithm. This first implementation was sufficient to obtain basin information for steepest ascent hillclimbing on the busy beaver problem that was described in §3.8.5(60). It suffered from the need to keep the entire basin in memory (though in the cases examined in [110], the basins never contained more than 10,000 vertices), and proved far slower than the third implementation.

The second implementation tried to reduce the memory requirements of the first implementation. The experiments of Chapter 4 occasionally required identifying basins that contained over half a million vertices. Instead of keeping all the vertices of the basin of attraction in memory simultaneously and updating the statistics of vertices that were re-encountered, this version simply wrote all its discoveries to a file that was post-processed once the run completed. The resulting file contained a line of output for each vertex that was encountered during the recursion. For example, the vertex 000 in Figure 49 on page 93 would appear as the subject of three output lines, as it would be encountered during descents from 100, 010 and 001. The post-processor sorted the output file by vertex identity and then collected the statistics for the groups of lines that corresponded to each vertex. The initial phase of this implementation required very little memory, and ran very quickly. Unfortunately, it produced output files containing many tens of megabytes, which rendered this approach less useful in practice than the first.

The first two solutions both employ a strategy that resembles a depth-first search in the landscape graph. Both of the difficulties with these solutions are a result of this form of exploration. A more appealing strategy is some form of breadth-first exploration of the landscape. A breadth-first exploration based on distance from the original vertex faces the problem that a vertex $v_1$ at distance $d_1$ might not appear to be in the basin of attraction of the original vertex until the depth-first search reaches some vertex $v_2$ at distance $d_2 > d_1$ and discovers that vertex $v_1$ has some probability of ascending to $v_2$. Such an algorithm would be faced with storage problems similar to that of the first implementation.

The breadth-first approach can be used however if the search proceeds breadth-first according to *fitness* rather than breadth-first according to distance from the initial vertex. If all the fitter neighbors of a vertex $v_1$ have been completely processed by the algorithm, and have all updated their downhill neighbors that could re-ascend, then the vertex $v_1$ can never again be encountered. Once the downhill neighbors of the vertex $v_1$ are updated according to their probabilities of reaching $v_1$, the statistics for $v_1$ can be written to a file (if these are to even be recorded) and the memory allocated to $v_1$ can be freed. This idea has several advantages:

- Vertices in the basin will be encountered multiple times, but will only be descended from *once*. In the previous implementations, each time a vertex was reached, its basin of attraction would be calculated afresh. This provides a significant increase in speed.

- This implementation returns memory to the operating system as it does not need to keep the entire basin of attraction in memory. There is a simple condition that indicates when a vertex has been completely dealt with, and at that time its summary statistics can be gathered and the vertex discarded from memory. The peak memory load is typically 15 to 20 percent less than that required by the first implementation.

- The vertices in the basin of attraction are found in order of decreasing fitness. This allows the efficient delineation of some fraction of the upper part of the basin of attraction. The earlier implementations also provide a method of approximation, as recursive calls can be made only if the probability of ascent (or the fitness) has not fallen below some lower limit.

This method provides a large speed increase and a moderate saving in memory. Its main drawback is the memory requirement, but it has been used to find basins containing over half a million vertices.

The data structures used to implement this version are naturally more involved. The two most important operations are the fast lookup of a vertex based on its identity and a fast method for retrieving the vertex with highest fitness from those that still need to be processed. The first task is best accomplished using a hash table and the second using a priority queue (implemented as a heap). I implemented a combination hash table and priority queue to solve both problems at once. The elements of the hash table contain a pointer into the priority queue and vice-versa. This allows $O(1)$ expected vertex identity comparisons to retrieve a vertex given its identity and $O(1)$ worst-case retrieval of the unprocessed vertex with the highest fitness (the priority queue retrieves a pointer into the hash table of the highest fitness vertex in constant time, and this is dereferenced in constant time since it points directly to the stored element and does not require hashing). Insertion of a new vertex requires insertion into both the hash table and the priority queue and this requires $O(\lg n)$ comparisons (worst and average case) where $n$ is the number of elements currently in the priority queue. Deletion requires $O(\lg n)$ comparisons (worst and average case) for similar reasons.

A somewhat simplified pseudo-code summary of this implementation is shown in Algorithm A.1. The pseudo-code assumes that the original point has already been inserted into both the hash table and the priority queue. Details of the coordination of pointers from the hash table to the priority queue and vice-versa are not shown. For more efficiency, the hash table find and insert can be done at the same time if the vertex is not present.

The above solution should not be taken as being in any way optimal. It is simply better than two others that are less sophisticated. A more efficient solution might employ a better mix of memory requirement and post-processing. If the hash table were allowed to contain only some maximum number of vertices, incomplete vertices (probably those that were judged most complete—for instance those with the highest ascent probabilities) could be written to a file for later coalescing, as in the second implementation. If a good balance could be found (and there is no reason to think this would not be straightforward, e.g., do as much in memory as you can afford), this would cap memory requirements and hopefully not demand too much disk space or post-processing time.

```
basin_size(initial_vertex)
VERTEX initial_vertex;
{
    PRIORITY_QUEUE pq;
    HASH_TABLE ht;
    VERTEX v, w;

    pq.insert(initial_vertex);
    ht.insert(initial_vertex);

    while (v = pq.get()) {
        for each (w|v ∈ N_φ(w) and f(w) <_F f(v)) {
            if (ht.find(w)) {
                update hashed entry for w;
            }
            else {
                pq.insert(w);
                ht.insert(w);
            }
        }

        free memory associated with v;
    }
}
```

**Algorithm A.1** A more sophisticated version of the reverse hillclimbing algorithm. This algorithm proceeds depth first (by fitness) from the original vertex. When all the vertices that could ascend to a vertex $v$ have been considered, the memory corresponding to $v$ can be freed.

APPENDIX B

# Balanced Hillclimbing

On the six landscapes examined in Chapter 4, SA had the highest probability of locating a peak on a single hillclimb on all of them. AA was always been second. Despite this, on five of the six problems, AA is the algorithm of choice since it expects to perform fewer evaluations per uphill step than SA does, and this advantage outweighs SA's greater location probability. It was always clear that, all other things being equal, making fewer evaluations per step rather than more would result in a better algorithm. Reverse hillclimbing shows that, at least on the instances of the problems considered in Chapter 4, all other things being equal, being more exploitative rather than less will also produce a better algorithm. Since SA and AA represent two extremes when it comes to these choices in hillclimber design, it is reasonable to expect that a more balanced algorithm might perform better than either of them.

This line of reasoning leads to the idea of hillclimbers that spend some amount of time identifying uphill neighbors and then move to the highest of those found. If we restrict attention to those hillclimbers that do not restart unless they have examined all neighbors and found no improvement, these algorithms can be thought of as members of a family of hillclimbers that make use of a function called *another*. The *another* function returns a boolean value which is used to decide whether another uphill neighbor should be sought. The arguments to the function are the number of neighbors ($n$), the number of uphill directions located so far ($u$), and the number of neighbors that have been tried ($t$). The assumption that these hillclimbers will always continue to search for an uphill neighbor until the first is found is equivalent to saying that $another(n, 0, t) = \text{TRUE}$ for all algorithms in this family. Similarly, we can assume that the *another* function will not be called if $t = n$, i.e., if all neighbors have been tried.

Both AA and SA are members of this family (LA and MA are not since they do not, in general, move to the steepest of the uphill neighbors). In AA, the *another* function always returns FALSE, never continuing the search for uphill neighbors after the first has been located. In SA, the *another* function always returns TRUE, since SA requires the examination of all neighbors. Within this family of hillclimbers, it is easy to see the sense in which AA and SA are the extreme algorithms.

There are several obvious alternatives for the *another* function. A very simple one is a function that returns TRUE only if $u < k$ for some $k$ (a constant or a function of $n$). This might be called Best-of-$k$ hillclimbing since the resulting algorithm finds

$k$ uphill neighbors (if possible) and moves to the best of these. I will denote this algorithm by BO-$k$. When $k = 1$, this algorithm becomes AA and when $k = n$ it becomes SA. AA will, on average, choose the median of all the uphill neighbors for a certain amount of work. On average, BO-2 will do twice as much work but will only choose an uphill neighbor whose rank is 1/3rd amongst all uphill neighbors. A rough estimate argues that twice as much work finding uphill neighbors will not be adequately compensated for by only receiving a one-sixth increase in the ranking of the uphill neighbor chosen.

Another strategy attempts to find another uphill neighbor only if less than $k$ neighbors have been examined for some $k$ (a constant or a function of $n$). That is, the *another* function returns TRUE if $t < k$. This algorithm also degenerates to AA and SA if $k = 1$ or $k = n$. When $k = 2$ the algorithm is very similar to AA, and can be thought of as identical to AA, except it seeks a second neighbor only on those occasions when AA found an uphill neighbor on the first trial. If $k$ is small compared to $n$, this algorithm can be thought of as trying for a second neighbor when AA gets lucky. This algorithm will be called Try-$k$ hillclimbing.

**Table 33.** The *another* function for several hillclimbers. $n$ is the number of neighbors, $u$ is the number of uphill directions located so far, and $t$ is the number of neighbors tried so far.

| Hillclimber | $another(n, u, t) =$ TRUE if |
|---|---|
| AA | NEVER |
| SA | ALWAYS |
| Best-of-$k$ | $u < k$ |
| PA | $n - t \geq t/u$ |
| Pess | $n - t \geq 2t/u$ |
| Try-$k$ | $t < k$ |

A more sophisticated strategy, that is close to SA in behavior, is to form an estimate of the number of neighbors that will need to be examined to locate an additional uphill neighbor. If this is greater than the remaining number of unexamined neighbors, then return FALSE. This hillclimber attempts to find many uphill neighbors, but stops looking once it appears likely that no more exist. It is similar to SA, but should use fewer evaluations, at the risk of missing uphill directions. For example, if only a single uphill direction has been found and more than half the neighbors have been examined, the *another* function of this algorithm would return FALSE. If two uphill neighbors have been found and more than two-thirds of the neighbors have been examined, it returns FALSE etc. The *another* function returns TRUE if $n - t \geq t/u$. This will be called *predictive ascent* hillclimbing, and abbreviated PA. A similar algorithm that is more pessimistic, denoted Pess, returns TRUE if $n - t \geq 2t/u$. The *another* functions for AA, SA, and the above algorithms are summarized in Table 33.

A graphical interpretation of these functions is shown in Figure 107. Any partitioning of such graphs into TRUE and FALSE regions represents a hillclimbing algorithm. These graphs also illustrate what is meant by the claim that AA and SA are in some sense algorithms at the two extremes of *another* functions.

Tables 34 to 39 show the result of hillclimbing with AA, SA, BO-2, PA, Pess,

**Table 34.** The results from 10,000 hillclimbs on a 16,12 NK landscape. The table shows, for eight hillclimbers, how frequently the best 5% of 1,000 randomly located peaks were located, and the mean and standard deviation of the number of evaluations to do so.

| | AA | SA | BO-2 | PA | Pess | Try-2 | Try-4 | Try-$n/2$ |
|---|---|---|---|---|---|---|---|---|
| Peaks found | 653 | 760 | 738 | 741 | 674 | 604 | 639 | 642 |
| Evals/peak | 492 | 646 | 548 | 579 | 596 | 540 | 537 | 597 |
| Evals/peak s.d. | 504 | 620 | 541 | 593 | 579 | 514 | 523 | 604 |

Try-2, Try-4 and Try-$n/2$ on the NK landscapes and busy beaver problems studied in Chapter 4. A † sign is used to indicate occasions when one of the six new hillclimbers has a mean number of evaluations per peak that is less than both AA and SA. The main conclusion that can be drawn from these tables is that AA is difficult to beat. Apart from one instance in which Pess beats AA (2-state busy beaver), the only hillclimbers to occasionally beat AA are the Try-$n$ hillclimbers. There is a correspondence between the difficulty of the problem and how Try-2, Try-4 and Try-$n/2$ compare to AA. As problems get simpler, the performance of Try-$n$ increases as $n$ does. As problems get harder, the expected number of uphill directions can be expected to fall and so an algorithm that spends less time looking for things that

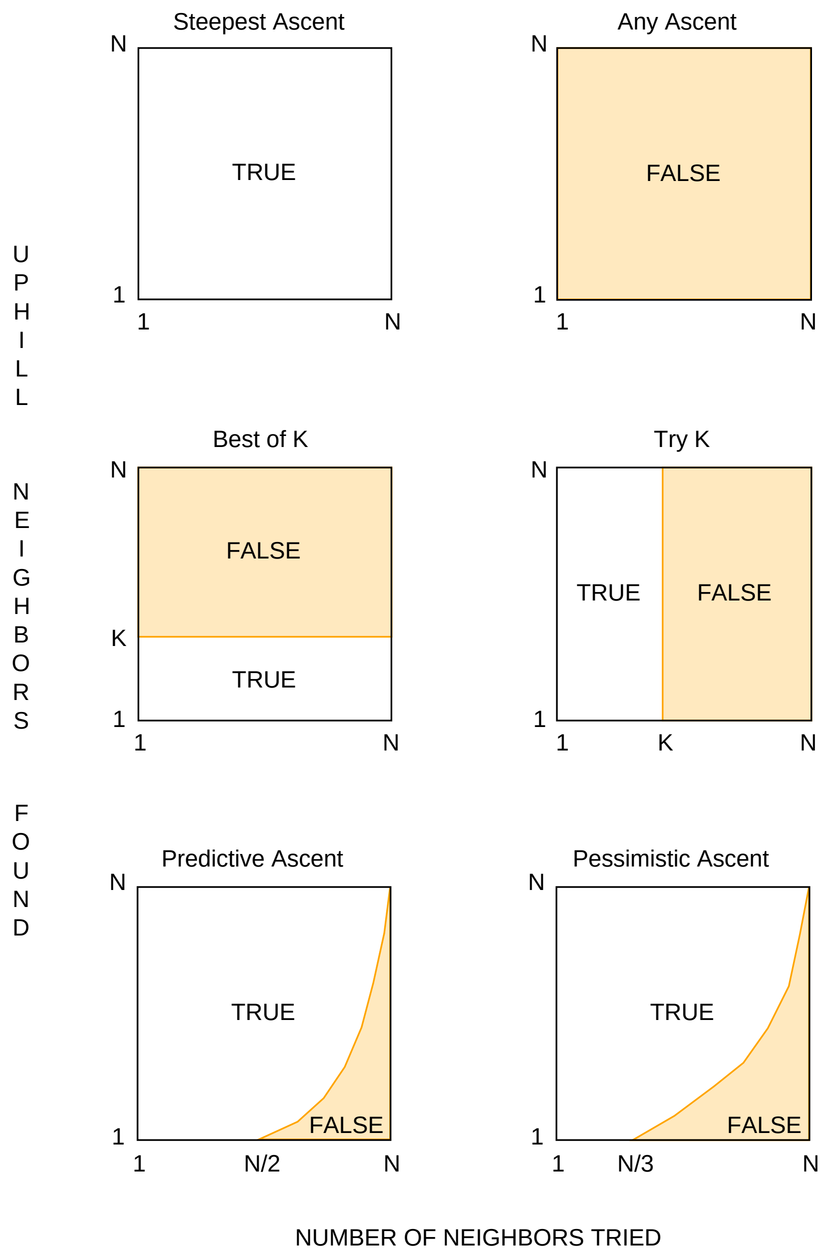


**Figure 107.** A graphical interpretation of six *another* functions. The X axis displays the number of neighbors examined and the Y axis the number of uphill neighbors found. The TRUE and FALSE regions indicate what the *another* function will return. All axes have a minimum of one and a maximum of $n$ (the number of neighbors).

probably do not exist will tend to perform better. It should be remembered that AA is actually Try-1. Try-2 is very similar to AA—it only looks for another uphill direction if one was found on the very first neighbor examined. As the number of uphill neighbors increases, spending more time looking for them may be worthwhile. For this reason, the Try-$n$ hillclimbers tend to do better on easy problems when $n$ is greater.

**Table 35.** The results from 10,000 hillclimbs on a 16,8 NK landscape. The table shows, for eight hillclimbers, how frequently the best 5% of 700 randomly located peaks were located, and the mean and standard deviation of the number of evaluations to do so.

| | AA | SA | BO-2 | PA | Pess | Try-2 | Try-4 | Try-$n/2$ |
|---|---|---|---|---|---|---|---|---|
| Peaks found | 1661 | 1873 | 1771 | 1775 | 1693 | 1634 | 1614 | 1682 |
| Evals/peak | 227 | 332 | 276 | 303 | 289 | 235 | 251 | 280 |
| Evals/peak s.d. | 198 | 290 | 244 | 263 | 262 | 209 | 226 | 241 |

**Table 36.** The results from 10,000 hillclimbs on a 16,4 NK landscape. The table shows, for eight hillclimbers, how frequently the best 5% of the 180 peaks were located, and the mean and standard deviation of the number of evaluations to do so.

| | AA | SA | BO-2 | PA | Pess | Try-2 | Try-4 | Try-$n/2$ |
|---|---|---|---|---|---|---|---|---|
| Peaks found | 1600 | 1999 | 1941 | 1894 | 1798 | 1649 | 1724 | 1839 |
| Evals/peak | 269 | 397 | 297 | 358 | 335 | $268^{\dagger}$ | 276 | 311 |
| Evals/peak s.d. | 250 | 346 | 267 | 327 | 303 | 236 | 248 | 273 |

**Table 37.** The results from 40 million hillclimbs on the 4-state busy beaver problem. The table shows, for eight hillclimbers, how frequently an optimal Turing machine was located, and the mean and standard deviation of the number of evaluations to do so (rounded to the nearest million).

| | AA | SA | BO-2 | PA | Pess | Try-2 | Try-4 | Try-$n/2$ |
|---|---|---|---|---|---|---|---|---|
| Peaks found | 44 | 94 | 68 | 51 | 59 | 41 | 44 | 58 |
| Evals/peak ($\times 10^6$) | 73 | 58 | 69 | 84 | 64 | 77 | 75 | 70 |
| Evals/peak s.d. ($\times 10^6$) | 66 | 56 | 63 | 95 | 67 | 66 | 66 | 76 |

**Table 38.** The results from 100,000 hillclimbs on the 3-state busy beaver problem. The table shows, for eight hillclimbers, how frequently an optimal Turing machine was located, and the mean and standard deviation of the number of evaluations to do so.

| | AA | SA | BO-2 | PA | Pess | Try-2 | Try-4 | Try-$n/2$ |
|---|---|---|---|---|---|---|---|---|
| Peaks found | 397 | 444 | 424 | 435 | 411 | 430 | 433 | 438 |
| Evals/peak | 12,670 | 17,732 | 16,607 | 14,527 | 14,013 | $11,761^{\dagger}$ | $11,903^{\dagger}$ | 13,694 |
| Evals/peak s.d. | 13,048 | 16,977 | 16,165 | 13,967 | 14,090 | 10,494 | 10,942 | 13,507 |

**Table 39.** The results from 100,000 hillclimbs on the 2-state busy beaver problem. The table shows, for eight hillclimbers, how frequently an optimal Turing machine was located, and the mean and standard deviation of the number of evaluations to do so.

| | AA | SA | BO-2 | PA | Pess | Try-2 | Try-4 | Try-$n/2$ |
|---|---|---|---|---|---|---|---|---|
| Peaks found | 3409 | 3795 | 3902 | 3991 | 3908 | 3791 | 3738 | 3874 |
| Evals/peak | 767 | 1003 | 918 | 786 | $743^{\dagger}$ | $718^{\dagger}$ | $742^{\dagger}$ | 777 |
| Evals/peak s.d. | 739 | 939 | 872 | 745 | 707 | 671 | 728 | 748 |

APPENDIX C

# Standard Errors from Chapter 3.

The following tables show the standard errors for all data plotted in Figures 13 to 43 in Chapter 3. Algorithm abbreviations appear in Table 1 on page 56. The various levels of achievement are listed across the top of each table. When an algorithm did not reach a performance level at least ten times, the table entry is left blank. In some cases, the levels displayed correspond only to the higher end of the performance range.

**Table 40.** The standard errors for CH, GA-S, GA-E and BH on the one max problem with 60 bits for the data graphed in Figure 13 on page 67.

| | 33 | 36 | 39 | 42 | 45 | 48 | 51 | 54 | 57 | 60 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH | 0·16 | 0·32 | 0·53 | 0·85 | 1·34 | 2·08 | 3·38 | 5·63 | 10·4 | 34·0 |
| GA-S | 0·05 | 0·19 | 1·06 | 7·49 | 12·9 | 13·9 | 16·0 | 19·2 | 24·6 | 44·6 |
| GA-E | 0·04 | 0·17 | 0·98 | 6·51 | 10·8 | 12·1 | 14·0 | 16·6 | 20·6 | 35·3 |
| BH | 0·10 | 0·12 | 0·14 | 0·15 | 0·17 | 0·20 | 0·24 | 0·31 | 0·43 | 2·45 |

**Table 41.** The standard errors for CH, GA-S, GA-E and BH on the one max problem with 120 bits for the data graphed in Figure 14 on page 67.

| | 66 | 72 | 78 | 84 | 90 | 96 | 102 | 108 | 114 | 120 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH | 0·37 | 0·74 | 1·36 | 2·45 | 4·41 | 7·54 | 12·8 | 22·8 | 45·2 | 1014 |
| GA-S | 0·13 | 1·30 | 14·5 | 20·8 | 24·6 | 29·7 | 38·9 | 58·1 | 901 | |
| GA-E | 0·08 | 0·77 | 8·80 | 12·6 | 14·6 | 18·0 | 23·0 | 32·6 | 82·4 | |
| BH | 0·22 | 0·27 | 0·30 | 0·33 | 0·36 | 0·43 | 0·50 | 0·64 | 0·93 | 7·62 |

**Table 42.** The standard errors for CH, GA-S, GA-E and BH on the fully easy problem with 10 subproblems for the data graphed in Figure 15 on page 68.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH | 0·65 | 3·79 | 8·46 | 13·8 | 20·2 | 29·7 | 46·0 | 92·7 | 301 | 2137 |
| GA-S | 0·06 | 1·02 | 16·4 | 48·6 | 85·0 | 202 | 753 | 5099 | 92126 | 1·0E7 |
| GA-E | 0·06 | 1·01 | 16·0 | 42·9 | 68·9 | 118 | 335 | 1435 | 14537 | 4·3E5 |
| BH | 1·02 | 3·24 | 12·7 | 80·0 | 614 | 9213 | | | | |

**Table 43.** The standard errors for CH, GA-S, GA-E and BH on the fully easy problem with 15 subproblems for the data graphed in Figure 16 on page 68.

| | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH | 27·5 | 35·4 | 46·2 | 73·8 | 152 | 401 | 1494 | 9610 | 1·4E5 | 5·0E6 |
| GA-S | 117 | 320 | 1451 | 11743 | 2·3E5 | 1·5E7 | | | | |
| GA-E | 121 | 303 | 1028 | 5838 | 53218 | 1·0E6 | | | | |
| BH | 373 | 2897 | 36718 | | | | | | | |

**Table 44.** The standard errors for CH, GA-S, GA-E and BH on the fully deceptive problem with 10 subproblems for the data graphed in Figure 17 on page 70.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH | 0·85 | 3·71 | 15·8 | 47·0 | 101 | 227 | 774 | 4710 | 69912 | 3·9E6 |
| GA-S | 0·06 | 1·00 | 10·5 | 44·1 | 201 | 761 | 4164 | 47627 | 1·1E6 | |
| GA-E | 0·06 | 1·00 | 9·09 | 36·5 | 152 | 488 | 2160 | 14596 | 2·2E5 | |
| BH | 20·2 | 491 | 20146 | | | | | | | |

**Table 45.** The standard errors for CH, GA-S, GA-E and BH on the fully deceptive problem with 15 subproblems for the data graphed in Figure 18 on page 70.

| | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH | 5·10 | 16·5 | 41·8 | 87·0 | 173 | 384 | 1222 | 5585 | 46369 | 4·9E5 |
| GA-S | 5·63 | 13·5 | 32·8 | 102 | 264 | 718 | 2826 | 19927 | 4·3E5 | |
| GA-E | 5·59 | 12·6 | 32·5 | 81·5 | 199 | 488 | 1438 | 6489 | 47247 | 6·6E5 |
| BH | 6213 | | | | | | | | | |

**Table 46.** The standard errors for CH, GA-S, GA-E and BH on the distributed fully deceptive problem with 10 subproblems for the data graphed in Figure 19 on page 71.

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| CH | 0·30 | 10·0 | 296 | 7479 | 3·8E5 | |
| GA-S | 0·04 | 0·77 | 182 | 17368 | 3·1E6 | |
| GA-E | 0·06 | 1·01 | 194 | 7705 | 2·8E5 | 7·2E6 |
| BH | 21·3 | 510 | 23629 | | | |

**Table 47.** The standard errors for CH, GA-S, GA-E and BH on the distributed fully deceptive problem with 15 subproblems for the data graphed in Figure 20 on page 71.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| CH | 0·15 | 1·92 | 65·2 | 682 | 6552 | 1·2E5 | 2·2E6 |
| GA-S | 0·04 | 0·44 | 20·3 | 1208 | 76983 | 6·1E6 | |
| GA-E | 0·04 | 0·44 | 18·1 | 665 | 10004 | 1·3E5 | 3·7E6 |
| BH | 18·0 | 248 | 4167 | | | | |

**Table 48.** The standard errors for CH, GA-S, GA-E and BH on the busy beaver problem with 3 states for the data graphed in Figure 21 on page 72.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| CH | 0·05 | 0·10 | 0·38 | 1·24 | 3·65 | 10·4 | 154 |
| GA-S | 0·01 | 0·02 | 0·15 | 1·77 | 13·0 | 48·7 | 787 |
| GA-E | 0·01 | 0·03 | 0·16 | 1·86 | 12·4 | 39·7 | 616 |
| BH | 0·16 | 0·18 | 0·71 | 1·17 | 1·51 | 12·1 | 126 |

**Table 49.** The standard errors for CH, GA-S, GA-E and BH on the busy beaver problem with 4 states for the data graphed in Figure 22 on page 72.

| | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH | 1·93 | 4·96 | 11·2 | 34·9 | 131 | 917 | 12930 | 1·4E5 | 1·3E6 | 5·9E6 |
| GA-S | 5·31 | 19·5 | 50·0 | 257 | 2201 | 43098 | 3·3E5 | 6·0E6 | 1·1E7 | |
| GA-E | 6·59 | 22·4 | 50·3 | 167 | 951 | 16545 | 2·1E5 | 2·4E6 | 8·1E6 | |
| BH | 1·03 | 1·58 | 4·66 | 16·8 | 78·4 | 1003 | 25372 | 2·4E5 | 1·4E6 | |

**Table 50.** The standard errors for CH, GA-S, GA-E and BH on Holland's royal road problem with $k = 4$ for the data graphed in Figure 23 on page 73.

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| CH | 3·23 | 22·2 | 403 | 1·8E5 |
| GA-S | 0·15 | 37·5 | 2070 | |
| GA-E | 0·15 | 34·3 | 1321 | 3·1E6 |
| BH | 2345 | | | |

**Table 51.** The standard errors for CH, GA-S, GA-E and BH on Holland's royal road problem with $k = 6$ for the data graphed in Figure 24 on page 73.

| | 1 | 2 | 3 |
|---|---|---|---|
| CH | 0·86 | 33·8 | 38849 |
| GA-S | 0·03 | 29·0 | 13786 |
| GA-E | 0·03 | 31·5 | 12262 |
| BH | 359 | 7·4E5 | |

**Table 52.** The standard errors for CH, CH-1S and CH-NJ on the one max problem with 120 bits for the data graphed in Figure 25 on page 75.

| | 66 | 72 | 78 | 84 | 90 | 96 | 102 | 108 | 114 | 120 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH | 0·37 | 0·74 | 1·36 | 2·45 | 4·41 | 7·54 | 12·8 | 22·8 | 45·2 | 1014 |
| CH-1S | 0·31 | 0·80 | 1·81 | 3·52 | 6·19 | 10·4 | 19·4 | 34·5 | 69·1 | 627 |
| CH-NJ | 1·57 | 8·73 | 257 | | | | | | | |

**Table 53.** The standard errors for CH, CH-1S and CH-NJ on the fully easy problem with 15 subproblems for the data graphed in Figure 26 on page 75.

| | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH | 27·5 | 35·4 | 46·2 | 73·8 | 152 | 401 | 1494 | 9610 | 1·4E5 | 5·0E6 |
| CH-1S | 28·4 | 36·7 | 48·0 | 65·9 | 97·5 | 164 | 334 | 879 | 3385 | 35004 |
| CH-NJ | | | | | | | | | | |

**Table 54.** The standard errors for CH, CH-1S and CH-NJ on the fully deceptive problem with 15 subproblems for the data graphed in Figure 27 on page 75.

| | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH | 41·8 | 87·0 | 173 | 384 | 1222 | 5585 | 46369 | 4·9E5 | | |
| CH-1S | 42·1 | 86·3 | 161 | 312 | 708 | 2090 | 7764 | 41828 | 3·8E5 | 5·3E6 |
| CH-NJ | | | | | | | | | | |

**Table 55.** The standard errors for CH, CH-1S and CH-NJ on the distributed fully deceptive problem with 15 subproblems for the data graphed in Figure 28 on page 75.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| CH | 0·15 | 1·92 | 65·2 | 682 | 6552 | 1·2E5 | 2·2E6 |
| CH-1S | 0·11 | 1·47 | 60·5 | 757 | 7573 | 97328 | 1·8E6 |
| CH-NJ | 0·24 | 2·51 | 50·8 | 1068 | | | |

**Table 56.** The standard errors for CH, CH-1S and CH-NJ on the busy beaver problem with 4 states for the data graphed in Figure 29 on page 76.

| | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH | 1·93 | 4·96 | 11·2 | 34·9 | 131 | 917 | 12930 | 1·4E5 | 1·3E6 | 5·9E6 |
| CH-1S | 1·92 | 5·36 | 11·7 | 36·4 | 136 | 964 | 15697 | 1·2E5 | 5·6E5 | 1·7E6 |
| CH-NJ | 782 | 7718 | 1·0E5 | | | | | | | |

**Table 57.** The standard errors for CH, CH-1S and CH-NJ on Holland's royal road problem with $k = 4$ for the data graphed in Figure 30 on page 76.

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| CH | 3·23 | 22·2 | 403 | 1·8E5 |
| CH-1S | 1·93 | 21·1 | 258 | 14619 |
| CH-NJ | 6·35 | 14063 | | |

**Table 58.** The standard errors for GA-S and GA-RC on the one max problem with 120 bits for the data graphed in Figure 32 on page 79.

| | 72 | 77 | 82 | 87 | 92 | 97 | 102 | 107 | 112 | 117 |
|---|---|---|---|---|---|---|---|---|---|---|
| GA-S | 1·30 | 11·5 | 20·2 | 23·2 | 26·2 | 30·5 | 38·9 | 53·0 | 108 | 4·6E5 |
| GA-RC | 0·91 | 10·2 | 55·0 | 2641 | 5·6E6 | | | | | |

**Table 59.** The standard errors for GA-S and GA-RC on the fully easy problem with 15 subproblems for the data graphed in Figure 33 on page 79.

| | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|
| GA-S | 0·43 | 8·11 | 31·8 | 62·8 | 117 | 320 | 1451 | 11743 | 2·3E5 | 1·5E7 |
| GA-RC | 0·42 | 8·56 | 86·0 | 2762 | 3·6E5 | | | | | |

**Table 60.** The standard errors for GA-S and GA-RC on the fully deceptive problem with 15 subproblems for the data graphed in Figure 34 on page 79.

| | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|
| GA-S | 0·44 | 5·63 | 13·5 | 32·8 | 102 | 264 | 718 | 2826 | 19927 | 4·3E5 |
| GA-RC | 0·43 | 6·74 | 38·6 | 616 | 52192 | | | | | |

**Table 61.** The standard errors for GA-S and GA-RC on the distributed fully deceptive problem with 15 subproblems for the data graphed in Figure 35 on page 79.

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| GA-S | 0·04 | 0·44 | 20·3 | 1208 | 76983 | 6·1E6 |
| GA-RC | 0·04 | 0·44 | 7·06 | 124 | 7444 | 8·5E5 |

**Table 62.** The standard errors for GA-S and GA-RC on the busy beaver problem with 4 states for the data graphed in Figure 36 on page 80.

| | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|
| GA-S | 0·68 | 5·31 | 19·5 | 50·0 | 257 | 2201 | 43098 | 3·3E5 | 6·0E6 | 1·1E7 |
| GA-RC | 0·87 | 6·87 | 39·0 | 285 | 4407 | 50396 | 7·1E5 | 7·1E6 | | |

**Table 63.** The standard errors for GA-S and GA-RC on Holland's royal road problem with $k = 4$ for the data graphed in Figure 37 on page 80.

| | 1 | 2 | 3 |
|---|---|---|---|
| GA-S | 0·15 | 37·5 | 2070 |
| GA-RC | 0·16 | 65·7 | |

**Table 64.** The standard errors for CH-1S, GA-E, BH-MM and BH-DMM on the one max problem with 120 bits for the data graphed in Figure 38 on page 82.

| | 66 | 72 | 78 | 84 | 90 | 96 | 102 | 108 | 114 | 120 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH-1S | 0·31 | 0·80 | 1·81 | 3·52 | 6·19 | 10·4 | 19·4 | 34·5 | 69·1 | 627 |
| GA-E | 0·08 | 0·77 | 8·80 | 12·6 | 14·6 | 18·0 | 23·0 | 32·6 | 82·4 | |
| BH-MM | 0·11 | 0·22 | 0·46 | 0·92 | 1·83 | 3·43 | 6·03 | 10·6 | 21·7 | 111 |
| BH-DMM | 0·12 | 0·21 | 0·37 | 0·66 | 1·15 | 2·05 | 3·59 | 6·29 | 13·3 | 69·5 |

**Table 65.** The standard errors for CH-1S, GA-E, BH-MM and BH-DMM on the fully easy problem with 15 subproblems for the data graphed in Figure 39 on page 82.

| | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH-1S | 28·4 | 36·7 | 48·0 | 65·9 | 97·5 | 164 | 334 | 879 | 3385 | 35004 |
| GA-E | 121 | 303 | 1028 | 5838 | 53218 | 1·0E6 | | | | |
| BH-MM | 28·8 | 37·1 | 47·7 | 61·6 | 81·2 | 110 | 159 | 278 | 666 | 2630 |
| BH-DMM | 19320 | 1·7E5 | | | | | | | | |

**Table 66.** The standard errors for CH-1S, GA-E, BH-MM and BH-DMM on the fully deceptive problem with 15 subproblems for the data graphed in Figure 40 on page 83.

| | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH-1S | 86·3 | 161 | 312 | 708 | 2090 | 7764 | 41828 | 3·8E5 | 5·3E6 | |
| GA-E | 81·5 | 199 | 488 | 1438 | 6489 | 47247 | 6·6E5 | | | |
| BH-MM | 98·8 | 163 | 257 | 399 | 727 | 1445 | 3909 | 13855 | 78320 | 8·2E5 |
| BH-DMM | | | | | | | | | | |

**Table 67.** The standard errors for CH-1S, GA-E, BH-MM and BH-DMM on the distributed fully deceptive problem with 15 subproblems for the data graphed in Figure 41 on page 83.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| CH-1S | 0·11 | 1·47 | 60·5 | 757 | 7573 | 97328 | 1·8E6 |
| GA-E | 0·04 | 0·44 | 18·1 | 665 | 10004 | 1·3E5 | 3·7E6 |
| BH-MM | 0·42 | 88·1 | 383 | 2028 | 14548 | 1·9E5 | |
| BH-DMM | 1·77 | 345 | 1106 | 5213 | 28631 | | |

**Table 68.** The standard errors for CH-1S, GA-E, BH-MM and BH-DMM on the busy beaver problem with 4 states for the data graphed in Figure 42 on page 83.

| | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|
| CH-1S | 1·92 | 5·36 | 11·7 | 36·4 | 136 | 964 | 15697 | 1·2E5 | 5·6E5 | 1·7E6 |
| GA-E | 6·59 | 22·4 | 50·3 | 167 | 951 | 16545 | 2·1E5 | 2·4E6 | 8·1E6 | |
| BH-MM | 0·89 | 2·23 | 5·07 | 17·4 | 78·4 | 528 | 6948 | 1·0E5 | 6·3E5 | 2·0E6 |
| BH-DMM | 1·77 | 4·36 | 10·0 | 36·7 | 150 | 890 | 12761 | 1·9E5 | 9·2E5 | |

**Table 69.** The standard errors for CH-1S, GA-E, BH-MM and BH-DMM on Holland's royal road problem with $k = 4$ for the data graphed in Figure 43 on page 83.

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| CH-1S | 1·93 | 21·1 | 258 | 14619 | |
| GA-E | 0·15 | 34·3 | 1321 | 3·1E6 | |
| BH-MM | 0·66 | 11·9 | 93·4 | 911 | 29560 |
| BH-DMM | 6·42 | 452 | | | |

APPENDIX D

# Function Repetition and Invariance in FDC

As noted in §5.7.1.1(137) the correlation between fitness and distance is unchanged when multiple copies of a problem are concatenated to form a larger problem. This appendix proves that this will occur for any number of copies of any function. We will denote the set of (fitness, distance) pairs of the original "base" function by $S = (X, Y)$ with cardinality $n$. When two copies of this function are concatenated, the (fitness, distance) pairs of the composite function will be the set $(X_i + X_j, Y_i + Y_j)$ for $1 \leq i, j \leq n$ since the fitnesses and distances of the new function are simply the sums of the fitnesses and distances in the solutions to the two subproblems. In general, let

$$S_k = (X^{(k)}, Y^{(k)}) = \bigcup_{(X_i, Y_i) \in S} (X_1 + X_2 + \cdots + X_k, Y_1 + Y_2 + \cdots + Y_k).$$

The set of (fitness, distance) pairs $S_k$ is obtained when $k$ copies of the base set are concatenated. Henceforth, we will abandon mention of fitness and distance, since the concepts are irrelevant to the proof. Clearly, $S_1 = (X^{(1)}, Y^{(1)}) = S = (X, Y)$. $X$ and $Y$ will generally be used in preference to $X^{(1)}$ and $Y^{(1)}$. The notation $X_i^{(k)}$ will be used to represent a general element of $X^{(k)}$. A simple counting argument shows that $|X^{(k)}| = |Y^{(k)}| = n^k$. The correlation $r(P)$ of any set of ordered pairs $P = (A, B)$ is given by

$$r(P) = \frac{C_{A,B}}{\sigma_A \sigma_B}$$

where

$$C_{A,B} = \frac{1}{n} \sum_i (A_i - \overline{A})(B_i - \overline{B})$$

is the covariance of $A$ and $B$, $\overline{A}$ and $\overline{B}$ are the means of $A$ and $B$, and

$$\sigma_A = \sqrt{\frac{\sum_{i=1}^n (A_i - \overline{A})^2}{n}} \quad \text{and} \quad \sigma_B = \sqrt{\frac{\sum_{i=1}^n (B_i - \overline{B})^2}{n}}$$

are the standard deviations of $A$ and $B$.[1]

[1]The denominator in the formula for standard deviation is usually given as $n - 1$, not $n$. This is

**Lemma 1** *For any two sets $A$ and $B$ both of size $n$,*

$$\sum(A-\overline{A})(B-\overline{B}) = \sum AB - \frac{\sum A \sum B}{n}.$$

This is a well-known statistical identity. It can be proved as follows:

$$\begin{aligned}
\sum(A-\overline{A})(B-\overline{B}) &= \sum AB - \overline{B}\sum A - \overline{A}\sum B + n\overline{A}\,\overline{B} \\
&= \sum AB - n\overline{A}\,\overline{B} - n\overline{A}\,\overline{B} + n\overline{A}\,\overline{B} \\
&= \sum AB - n\overline{A}\,\overline{B} \\
&= \sum AB - \frac{\sum A \sum B}{n}.
\end{aligned}$$

**Lemma 2** *For all integers $k > 0$, $\sum_i X_i^{(k)} = kn^k\overline{X}$.*

The proof will rely on an identity that will be frequently used in subsequent proofs:

$$\sum_{1 \leq i \leq n^k} X_i^{(k)} = \sum_{\substack{1 \leq i \leq n^{k-1} \\ 1 \leq j \leq n}} (X_i^{(k-1)} + X_j)$$

An identical result holds for $Y^{(k)}$. The identity follows from the construction of $X^{(k)}$. This double summation will hereafter be denoted $\sum\limits_{ij}$ with the understanding that the index $i$ runs from 1 to $n^{k-1}$ and the index $j$ runs from 1 to $n$. The proof of the lemma follows:

$$\begin{aligned}
\sum_i X_i^{(k)} &= \sum_{i,j}(X_i^{(k-1)} + X_j) \\
&= \sum_{i,j} X_i^{(k-1)} + \sum_{i,j} X_j \\
&= n\sum_i X_i^{(k-1)} + n^{k-1}\sum_j X_j \\
&= n\sum_i X_i^{(k-1)} + n^k\overline{X}
\end{aligned}$$

---

because the value of the mean that is used in the formula is an *estimate* of the true mean. When the actual mean of the underlying distribution is known (as in the computation of FDC by examination of the entire representation space), $n$ replaces $n-1$ in the computation of standard deviation [192, page 19].

Letting $T_k = \sum_i X_i^{(k)}$, we have arrived at the recurrence relation

$$T_k = \begin{cases} nT_{k-1} + n^k\overline{X} & k > 1 \\ n\overline{X} & k = 1. \end{cases}$$

Small values of $k$ suggest that $T_k = kn^k\overline{X}$ which is quickly confirmed by induction on $k$.

**Corollary 1** *For all integers $k > 0$, $\overline{X^{(k)}} = k\overline{X}$.*

Since $|X^{(k)}| = n^k$ and $\sum_i X_i^{(k)} = kn^k\overline{X}$,

$$\overline{X^{(k)}} = \frac{kn^k\overline{X}}{n^k} = k\overline{X}.$$

**Theorem 1** *For all integers $k > 0$, $r(S_k) = r(S)$.*

This is the main result and its proof indicates that any number of copies of the base function $S$ will exhibit the same fitness distance correlation (FDC). The proof is in two parts. In the first, it is proved that

$$C_{X^{(k)},Y^{(k)}} = kC_{X,Y} \quad \forall\, k > 0, \tag{1}$$

and in the second that

$$\sigma_{X^{(k)}}\sigma_{Y^{(k)}} = k\sigma_X\sigma_Y \quad \forall\, k > 0. \tag{2}$$

It follows immediately that

$$r(S_k) = \frac{C_{X^{(k)},Y^{(k)}}}{\sigma_{X^{(k)}}\sigma_{Y^{(k)}}} = \frac{kC_{X,Y}}{k\sigma_X\sigma_Y} = \frac{C_{X,Y}}{\sigma_X\sigma_Y} = r(S).$$

**Lemma 3** *For all integers $k > 0$, $C_{X^{(k)},Y^{(k)}} = kC_{X,Y}$.*

$$\begin{aligned} C_{X^{(k)},Y^{(k)}} &= \frac{1}{n^k}\sum_i (X_i^{(k)} - \overline{X^{(k)}})(Y_i^{(k)} - \overline{Y^{(k)}}) \\ n^k C_{X^{(k)},Y^{(k)}} &= \sum_i (X_i^{(k)} - \overline{X^{(k)}})(Y_i^{(k)} - \overline{Y^{(k)}}) \\ &= \sum_i X_i^{(k)}Y_i^{(k)} - \frac{\sum_i X_i^{(k)} \sum_i Y_i^{(k)}}{n^k} \end{aligned}$$

$$
\begin{aligned}
&= \sum_i X_i^{(k)} Y_i^{(k)} - \frac{n^k k \overline{X} n^k k \overline{Y}}{n^k} \\
&= \sum_i X_i^{(k)} Y_i^{(k)} - n^k k^2 \overline{X}\,\overline{Y} \\
&= \sum_i (X_i^{(k)} Y_i^{(k)} - k^2 \overline{X}\,\overline{Y}) \\
&= \sum_{i,j} \left[ (X_i^{(k-1)} + X_j)(Y_i^{(k-1)} + Y_j) - k^2 \overline{X}\,\overline{Y} \right] \\
&= \sum_{i,j} \left[ X_i^{(k-1)} Y_i^{(k-1)} + Y_j X_i^{(k-1)} + X_j Y_i^{(k-1)} + X_j Y_j - k^2 \overline{X}\,\overline{Y} \right] \\
&= \sum_{i,j} \left( X_i^{(k-1)} Y_i^{(k-1)} - (k-1)^2 \overline{X}\,\overline{Y} \right) + \sum_j Y_j \sum_i X_i^{(k-1)} \\
&\quad + \sum_j X_j \sum_i Y_i^{(k-1)} + \sum_{i,j} X_j Y_j - \sum_{i,j} (2k-1) \overline{X}\,\overline{Y} \\
&= \sum_{i,j} (X_i^{(k-1)} Y_i^{(k-1)} - \overline{X^{(k-1)}}\,\overline{Y^{(k-1)}}) + n \overline{Y} n^{k-1} (k-1) \overline{X} \\
&\quad + n \overline{X} n^{k-1} (k-1) \overline{Y} + n^{k-1} \sum_i X_i Y_i - n^k (2k-1) \overline{X}\,\overline{Y} \\
&= \sum_{i,j} (X_i^{(k-1)} Y_i^{(k-1)} - \overline{X^{(k-1)}}\,\overline{Y^{(k-1)}}) + 2n^k (k-1) \overline{X}\,\overline{Y} \\
&\quad + n^{k-1} \sum_i X_i Y_i - n^k (2k-1) \overline{X}\,\overline{Y} \\
&= \sum_{i,j} (X_i^{(k-1)} Y_i^{(k-1)} - \overline{X^{(k-1)}}\,\overline{Y^{(k-1)}}) + n^k \overline{X}\,\overline{Y} (2(k-1) - (2k-1)) \\
&\quad + n^{k-1} \sum_i X_i Y_i \\
&= \sum_{i,j} (X_i^{(k-1)} Y_i^{(k-1)} - \overline{X^{(k-1)}}\,\overline{Y^{(k-1)}}) - n^k \overline{X}\,\overline{Y} + n^{k-1} \sum_i X_i Y_i \\
&= n \sum_i (X_i^{(k-1)} Y_i^{(k-1)} - \overline{X^{(k-1)}}\,\overline{Y^{(k-1)}}) + n^{k-1} (\sum_i X_i Y_i - n \overline{X}\,\overline{Y}) \\
&= n \sum_i (X_i^{(k-1)} Y_i^{(k-1)} - \overline{X^{(k-1)}}\,\overline{Y^{(k-1)}}) \\
&\quad + n^{k-1} (\sum_i X_i Y_i - \frac{\sum_i X_i \sum_i Y_i}{n}) \\
&= n \sum_i (X_i^{(k-1)} Y_i^{(k-1)} - \overline{X^{(k-1)}}\,\overline{Y^{(k-1)}}) \\
&\quad + n^{k-1} \sum_i (X_i - \overline{X})(Y_i - \overline{Y}) \qquad (3)
\end{aligned}
$$

We now examine the sum in the first term of (3), $\sum_i (X_i^{(k-1)} Y_i^{(k-1)} - \overline{X^{(k-1)}}\,\overline{Y^{(k-1)}})$.

If we let $w = k-1$, then

$$\begin{aligned}
\sum_i (X_i^{(w)} Y_i^{(w)} - \overline{X^{(w)}}\,\overline{Y^{(w)}}) &= \sum_i X_i^{(w)} Y_i^{(w)} - \sum_i \overline{X^{(w)}}\,\overline{Y^{(w)}} \\
&= \sum_i X_i^{(w)} Y_i^{(w)} - w^2 n^w \overline{X}\,\overline{Y} \\
&= \sum_i X_i^{(w)} Y_i^{(w)} - w^2 n^w \overline{X}\,\overline{Y} - w^2 n^w \overline{X}\,\overline{Y} + w^2 n^w \overline{X}\,\overline{Y} \\
&= \sum_i X_i^{(w)} Y_i^{(w)} - w\overline{Y} w n^w \overline{X} - w\overline{X} w n^w \overline{Y} + n^w w\overline{X} w\overline{Y} \\
&= \sum_i X_i^{(w)} Y_i^{(w)} - w\overline{Y} \sum_i X_i^{(w)} - w\overline{X} \sum_i Y_i^{(w)} \\
&\quad + \sum_i \overline{X^{(w)}}\,\overline{Y^{(w)}} \\
&= \sum_i (X_i^{(w)} Y_i^{(w)} - w\overline{Y} X_i^{(w)} - w\overline{X} Y_i^{(w)} + \overline{X^{(w)}}\,\overline{Y^{(w)}}) \\
&= \sum_i (X_i^{(w)} Y_i^{(w)} - \overline{Y^{(w)}} X_i^{(w)} - \overline{X^{(w)}} Y_i^{(w)} + \overline{X^{(w)}}\,\overline{Y^{(w)}}) \\
&= \sum_i (X_i^{(w)} - \overline{X^{(w)}})(Y_i^{(w)} - \overline{Y^{(w)}}) \qquad (4)
\end{aligned}$$

Replacing $w$ by $k-1$ and substituting (4) back into (3), we have

$$\begin{aligned}
n^k C_{X^{(k)},Y^{(k)}} &= n \sum_i (X_i^{(k-1)} - \overline{X^{(k-1)}})(Y_i^{(k-1)} - \overline{Y^{(k-1)}}) \\
&\quad + n^{k-1} \sum_i (X_i - \overline{X})(Y_i - \overline{Y}) \\
C_{X^{(k)},Y^{(k)}} &= \frac{\sum_i (X_i^{(k-1)} - \overline{X^{(k-1)}})(Y_i^{(k-1)} - \overline{Y^{(k-1)}})}{n^{k-1}} + \frac{\sum_i (X_i - \overline{X})(Y_i - \overline{Y})}{n} \\
&= C_{X^{(k-1)},Y^{(k-1)}} + C_{X,Y} \\
&= k C_{X,Y}
\end{aligned}$$

This proves (1), the first part of the main theorem.

**Lemma 4** *For all integers $k > 0$, $\sigma_{X^{(k)}} \sigma_{Y^{(k)}} = k \sigma_X \sigma_Y$*

This will be proved by examining $\sum_i \left(X_i^{(k)} - \overline{X^{(k)}}\right)^2$ from which $\sigma_X$ (and similarly $\sigma_Y$) are easily obtained.

$$\sum_i \left(X_i^{(k)} - \overline{X^{(k)}}\right)^2 = \sum_i \left(X_i^{(k)} - k\overline{X}\right)^2$$

$$
\begin{aligned}
&= \sum_{i,j} \Big[(X_i^{(k-1)} - (k-1)\overline{X}) + (X_j - \overline{X})\Big]^2 \\
&= \sum_{i,j} (X_i^{(k-1)} - (k-1)\overline{X})^2 + 2\sum_{i,j}(X_i^{(k-1)} - (k-1)\overline{X})(X_j - \overline{X}) \\
&\quad + \sum_{i,j} (X_j - \overline{X})^2 \qquad (5)
\end{aligned}
$$

Consider the middle term of (5):

$$
\begin{aligned}
2\sum_{i,j}(X_i^{(k-1)} - (k-1)\overline{X})(X_j - \overline{X}) &= 2\Big[\sum_{i,j} X_i^{(k-1)} X_j - \sum_{i,j} X_i^{(k-1)}\overline{X} \\
&\quad - \sum_{i,j}(k-1)\overline{X}X_j + \sum_{i,j}(k-1)\overline{X}^2\Big] \\
&= 2\Big[\sum_i X_i^{(k-1)} \sum_j X_j - n\overline{X}\sum_i X_i^{(k-1)} \\
&\quad - (k-1)n^{k-1}\overline{X}\sum_j X_j + (k-1)n^k\overline{X}^2\Big] \\
&= 2\Big[(k-1)n^{k-1}\overline{X}n\overline{X} - (k-1)n^k\overline{X}^2 \\
&\quad - (k-1)n^k\overline{X}^2 + (k-1)n^k\overline{X}^2\Big] \\
&= 0
\end{aligned}
$$

Therefore (5) becomes

$$
\begin{aligned}
\sum_i (X_i^{(k)} - \overline{X^{(k)}})^2 &= \sum_{i,j} (X_i^{(k-1)} - (k-1)\overline{X})^2 + \sum_{i,j} (X_j - \overline{X})^2 \\
&= n\sum_i (X_i^{(k-1)} - (k-1)\overline{X})^2 + n^{k-1}\sum_j (X_j - \overline{X})^2 \\
&= n\sum_i (X_i^{(k-1)} - \overline{X^{(k-1)}})^2 + n^{k-1}\sum_j (X_j - \overline{X})^2
\end{aligned}
$$

Letting $T_k = \sum_i (X_i^{(k)} - \overline{X^{(k)}})^2$, we have arrived at the recurrence relation

$$
T_k = \begin{cases} nT_{k-1} + n^{k-1}T_1 & k > 1 \\ \sum_i (X_i - \overline{X})^2 & k = 1. \end{cases}
$$

Small values of $k$ suggest that $T_k = kn^{k-1}\sum_i (X_i - \overline{X})^2$ which is quickly confirmed by induction on $k$. Thus we have

$$\begin{aligned}
\sigma_{X^{(k)}} &= \sqrt{\frac{\sum_i \left(X_i^{(k)} - \overline{X^{(k)}}\right)^2}{n^k}} \\
&= \sqrt{\frac{kn^{k-1}\sum_i \left(X_i - \overline{X}\right)^2}{n^k}} \\
&= \sqrt{k}\sqrt{\frac{\sum_i \left(X_i - \overline{X}\right)^2}{n}} \\
&= \sqrt{k}\sigma_X
\end{aligned}$$

Similarly, $\sigma_{Y^{(k)}} = \sqrt{k}\sigma_Y$. Therefore $\sigma_{X^{(k)}}\sigma_{Y^{(k)}} = k\sigma_X\sigma_Y$ which proves (2) and completes the proof of the main theorem.

REFERENCES

[1] A. Desmond and J. Moore. *Darwin.* Penguin, London, 1992.

[2] R. Chambers. *Vestiges of the Natural History of Creation.* Chicago University Press, Chicago IL, 1994.

[3] J. Huxley. *Evolution, The Modern Synthesis.* Allen and Unwin, London, 1942.

[4] E. Mayr. *The Growth of Biological Thought.* Harvard University Press, Cambridge, MA, 1982.

[5] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, Reading, MA, 1989.

[6] D. B. Fogel. An introduction to simulated evolutionary optimization. *IEEE Transactions on Neural Networks*, 5(1):3–14, Jan 1994. Special Issue on Evolutionary Computation.

[7] C. G. Langton. Artificial life. In C. G. Langton, editor, *Artificial Life*, volume 1, pages 1–47, Santa Fe, NM, 1989. Santa Fe Institute Studies in the Sciences of Complexity., Addison-Wesley.

[8] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence Through Simulated Evolution.* John Wiley and Sons, New York, 1966.

[9] J. H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, MI, 1975.

[10] I. Rechenberg. *Evolutionsstrategie: Optimierung Techniquer Systeme nach Prinzipien der Biologischen Evolution.* Frommann-Holzboog Verlag, Stuttgart, 1973.

[11] H.-P. Schwefel. Numerische optimierung von computer-modellen mittels der evolutionsstrategie. *Interdisciplinary systems research*, 26, 1977. Birkhäuser, Basel.

[12] T. Bäck and H.-P. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–24, 1993.

[13] D. E. Goldberg, K. Deb, and B. Korb. Don't worry, be messy. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 24–30, San Mateo, CA, 1991. Morgan Kaufmann.

[14] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT Press, Cambridge, MA, 1992.

[15] J. R. Koza. *Genetic Programming II: Automatic Discovery of Reusable Programs.* MIT Press, Cambridge, MA, 1994.

[16] S. Wright. The roles of mutation, inbreeding, crossbreeding and selection in evolution. In *Proceedings of the sixth international congress of genetics*, volume 1, pages 356–366, 1932.

[17] J. B. S. Haldane. A mathematical theory of natural and artificial selection, part viii: Metastable populations. *Transactions of the Cambridge Philosophical Society*, 27:137–142, 1931.

[18] N. Eldredge. *Macroevolutionary Dynamics: Species, Niches and Adaptive Peaks.* McGraw-Hill, 1989.

[19] G. G. Simpson. *Tempo and Mode in Evolution.* Columbia University Press, New York, 1944.

[20] S. Wright. The distribution of gene frequencies in populations. In *Proceedings of the National Academy of Science*, volume 23, pages 307–320, 1937.

[21] R. Lande. Natural selection and random genetic drift in phenotype evolution. *Evolution*, 30:314–334, 1976.

[22] D. Sherrington and S. Kirkpatrick. Solvable model of a spin glass. *Physical Review Letters*, 32:1792–1796, 1975.

[23] C. Amitrano, L. Peliti, and M. Saber. A spin-glass model of evolution. In A. Perelson and S. A. Kauffman, editors, *Molecular Evolution on Rugged Landscapes: Proteins, RNA and the Immune System*, pages 27–38, Redwood City, CA, Dec 1987. Santa Fe Institute Studies in the Sciences of Complexity, volume IX, Addison-Wesley.

[24] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Computer Algorithms.* Addison-Wesley, Reading, MA, 1974.

[25] M. Minsky. Steps toward artificial intelligence. In E. A. Feigenbaum and J. Friedman, editors, *Computers and Thought*, pages 406–450. McGraw-Hill, 1963. This originally appeared in *Proc, IRE*, vol. 49, pp. 8–30, 1961.

[26] D. H. Ackley. *A Connectionist Machine for Genetic Hillclimbing.* Kluwer Academic Publishers, Boston, MA, 1987.

[27] G. J. E. Rawlins. Introduction. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, volume 1, pages 1–10. Morgan Kaufmann, San Mateo, CA, 1991.

[28] J. C. Culberson and G. J. E. Rawlins. Genetic algorithms as function optimizers. Unpublished manuscript., December 1992.

[29] J. C. Culberson. Mutation-crossover isomorphisms and the construction of discriminating functions. *Evolutionary Computation*, 2(3):279–311, 1995.

[30] K. A. De Jong. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems.* PhD thesis, University of Michigan, 1975. Dissertation Abstracts International 36(10), 5410B. (University Microfilms No. 76–9381).

[31] J. Pearl. *Heuristics: Intelligent Search Strategies for Computer Problem Solving.* Addison-Wesley, Reading, MA, 1984.

[32] S. Amarel. On representations of problems of reasoning about actions. In D. Michie, editor, *Machine Intelligence*, volume 3, pages 131–171. Edinburgh University Press, 1968.

[33] R. E. Korf. Macro-Operators: A weak method for learning. *Artificial Intelligence*, 26:35–77, 1985.

[34] T. Shiple, P. Kollaritsch, D. J. Smith, and J. Allen. Area evaluation metrics for transistor placement. In *Proceedings of the IEEE International Conference on Computer Design: VLSI Computers and Processors*, pages 428–433, Washington, DC, October 1988. Computer Society of the IEEE.

[35] C. A. R. Hoare. Quicksort. *Computer Journal*, 5(1):10–15, 1962.

[36] M. Mitchell, S. Forrest, and J. H. Holland. The royal road for genetic algorithms: Fitness landscapes and GA performance. In F. J. Varela and P. Bourgine, editors, *Proceedings of the First European Conference on Artificial Life. Toward a Practice of Autonomous Systems*, pages 245–254, Cambridge, MA, 11–13 Dec 1992. MIT Press.

[37] S. Wright. Surfaces of selective value revisited. *American Naturalist*, 131(1):115–123, 1988.

[38] T. C. Jones and S. Forrest. Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In L. J. Eshelman, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, 1995. (to appear).

[39] T. C. Jones and S. Forrest. Genetic algorithms and heuristic search. Technical Report 95–02–021, Santa Fe Institute, Santa Fe, NM, February 1995. Available via ftp from ftp.santafe.edu in pub/terry/gahs.ps.gz.

[40] D. E. Knuth. *The Art of Computer Programming*, volume 2 : Sorting and Searching. Addison-Wesley, Reading, MA, 2nd edition, 1980.

[41] P. Kanerva. *Sparse Distributed Memory*. Bradford Books, MIT Press, Cambridge MA, 1988.

[42] F. Harary, R. Z. Norman, and D. Cartwright. *Structural Models: An Introduction to the Theory of Directed Graphs*. John Wiley & Sons, New York, 1965.

[43] N. J. Nilsson. *Problem-Solving Methods in Artificial Intelligence*. McGraw-Hill, 1971.

[44] E. Rich. *Artificial Intelligence*. McGraw-Hill, New York, 1983.

[45] R. B. Banerji. *Theory of problem Solving: An Approach to Artificial Intelligence*. Modern Analytic and Computational Methods in Science and Mathematics. American Elsevier, New York, NY, 1969.

[46] D. J. Smith, February 1995. Personal communication.

[47] G. Liepins and M. D. Vose. Representational issues in genetic optimization. *Journal of Experimental and Theoretical Artificial Intelligence*, 2:101–115, 1990.

[48] K. Mathias and L. D. Whitley. Remapping hyperspace during genetic search: Canonical delta folding. In L. D. Whitley, editor, *Foundations of Genetic Algorithms*, volume 2, pages 167–186, San Mateo, CA, 1993. Morgan Kaufmann.

[49] N. N. Schraudolph and R. K. Belew. Dynamic parameter encoding for genetic algorithms. *Machine Learning*, 9(1):9–21, June 1992.

[50] C. G. Shaefer. The ARGOT strategy: Adaptive representation genetic optimizer technique. In J. J. Grefenstette, editor, *Proceedings of the Second International Conference on Genetic Algorithms*, pages 50–55, Hillsdale NJ, 1987. Lawrence Erlbaum Associates.

[51] L. D. Whitley, K. Mathias, and P. Fitzhorn. Delta coding: An iterative search strategy for genetic algorithms. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 77–84, San Mateo, CA, 1991. Morgan Kaufmann.

[52] A. Nix and M. D. Vose. Modeling genetic algorithms with Markov chains. *Annals of Mathematics and Artificial Intelligence*, 5:79–88, 1992.

[53] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, May 1983.

[54] D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon. Optimization by simulated annealing: An experimental evaluation; Part I, graph partitioning. *Operations Research*, 37(6):865–892, 1989.

[55] D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon. Optimization by simulated annealing: An experimental evaluation; Part II, graph coloring and number partitioning. *Operations Research*, 39(3):378–406, 1991.

[56] T. Bäck, F. Hoffmeister, and H.-P. Schwefel. A survey of evolution strategies. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 2–9, San Mateo, CA, 1991. Morgan Kaufmann.

[57] K. E. Kinnear Jr., editor. *Advances in Genetic Programming*, Cambridge, MA, 1994. MIT Press.

[58] I. M. Oliver, D. J. Smith, and J. R. C. Holland. A study of permutation crossover operators on the traveling salesman problem. In J. J. Grefenstette, editor, *Genetic Algorithms and their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, pages 224–230, Hillsdale, NJ, 1987. Lawrence Erlbaum.

[59] L. D. Whitley, T. Starkweather, and F. D'Ann. Scheduling problems and traveling salesmen: The genetic edge recombination operator. In J. D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 133–140, San Mateo, CA, June 4–7 1989. Morgan Kaufmann.

[60] B. Manderick, M. De Weger, and P. Spiessens. The genetic algorithm and the structure of the fitness landscape. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 143–150, San Mateo, CA, 1991. Morgan Kaufmann.

[61] K. Mathias and L. D. Whitley. Genetic operators, the fitness landscape and the traveling salesman problem. In R. Männer and B. Manderick, editors, *Parallel Problem Solving From Nature*, volume 2, pages 219–228, Amsterdam, The Netherlands, 1992. Elsevier Science Publishers B.V.

[62] P. F. Stadler and W. Schnabl. The landscape of the traveling salesman problem. *Physics Letters A*, 161:337–344, 1992.

[63] E. D. Weinberger. Fourier and taylor series on fitness landscapes. *Biological Cybernetics*, 65:321–330, 1990.

[64] E. D. Weinberger. Measuring correlations in energy landscapes and why it matters. In H. Atmanspacher and H. Scheingraber, editors, *Information Dynamics*, pages 185–193. Plenum Press, New York, 1991.

[65] E. D. Weinberger. Correlated and uncorrelated fitness landscapes and how to tell the difference. *Biological Cybernetics*, 63:325–336, 1990.

[66] R. Parsons, S. Forrest, and C. Burks. Genetic operators for the DNA fragment assembly problem. *Machine Learning*, 1995. (in press).

[67] N. J. Radcliffe and P. D. Surry. Fitness variance of formae and performance prediction. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms*, volume 3, San Mateo, CA, 1995. Morgan Kaufmann.

[68] D. E. Goldberg and R. Lingle. Alleles, loci, and the traveling salesman problem. In J. J. Grefenstette, editor, *Proceedings of an International Conference on Genetic Algorithms and their Applications*, pages 154–159. Lawrence Erlbaum, Hillsdale, NJ, 24–26 July 1985.

[69] T. Starkweather, S. McDaniel, K. Mathias, and L. D. Whitley. A comparison of genetic sequencing operators. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 69–76, San Mateo, CA, 1991. Morgan Kaufmann.

[70] J. Dzubera and L. D. Whitley. Advanced correlation analysis of operators for the traveling salesman problem. In Y. Davidor, H.-P. Schwefel, and R. Männer, editors, *Parallel Problem Solving From Nature – PPSN III, volume 866 of Lecture Notes in Computer Science*, pages 68–77, Berlin, 1994. Springer-Verlag.

[71] M. Gorges-Schleuter. ASPARAGOS An asynchronous parallel genetic optimization strategy. In J. D. Schaffer, editor, *Proceedings of the Third International*

*Conference on Genetic Algorithms*, pages 422–427, San Mateo, CA, June 4–7 1989. Morgan Kaufmann.

[72] H. Mühlenbein. Evolution in time and space – The parallel genetic algorithm. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, volume 1, pages 316–337, San Mateo, CA, 1991. Morgan Kaufmann.

[73] P. Moscato and M. G. Norman. A "memetic" approach for the travelling salesman problem—implementation of a computational ecology for combinatorial optimisation on message-passing systems. In *Proceedings of the International Conference on Parallel Computing and Transputer Applications*, Amsterdam, 1992. IOS Press.

[74] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag, New York, 2nd edition, 1994.

[75] K. A. De Jong. Genetic algorithms; A 10 year perspective. In J. J. Grefenstette, editor, *Proceedings of an International Conference on Genetic Algorithms and their Applications*, pages 169–177, Hillsdale, NJ, 24–26 July 1985. Carnegie Mellon University, Lawrence Erlbaum.

[76] S. Kirkpatrick. Optimization by simulated annealing: Quantitative studies. *Journal of Statistical Physics*, 34:975–986, 1984.

[77] J. J. Hopfield and D. Tank. Neural computation of decisions in optimization problems. *Biological Cybernetics*, 52:141–152, 1985.

[78] J. T. Richardson, M. R. Palmer, G. Liepins, and M. Hilliard. Some guidlines for genetic algorithms with penalty functions. In J. D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 191–197, San Mateo, CA, June 4–7 1989. Morgan Kaufmann.

[79] W. Siedlecki and J. Sklansky. Constrained genetic optimization via dynamic reward-penalty balancing and its use in pattern recognition. In J. D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 141–150, San Mateo, CA, June 4–7 1989. Morgan Kaufmann.

[80] L. Davis and M. Steenstrup. Genetic algorithms and simulated annealing: An overview. In L. Davis, editor, *Genetic Algorithms and Simulated Annealing*, pages 1–11. Pitman (Morgan Kaufmann), London, 1987.

[81] D. Orvosh and L. Davis. Shall we repair? Genetic algorithms, combinatorial optimization and feasibility constraints. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, page 650, San Mateo, CA, 1993. Morgan Kaufmann.

[82] W. Banzhaf. Genetic programming for pedestrians. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, page 628, San Mateo, CA, 1993. Morgan Kaufmann.

[83] R. Nakano and T. Yamada. Conventional genetic algorithm for job shop problems. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 474–479, San Mateo, CA, 1991. Morgan Kaufmann.

[84] J. C. Bean. Genetics and random keys for sequencing and optimization. *ORSA Journal on Computing*, 6:154–160, 1994.

[85] D. L. Battle and M. D. Vose. Isomorphisms of genetic algorithms. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, volume 1, pages 242–251, San Mateo, CA, 1991. Morgan Kaufmann.

[86] J. Horn, D. E. Goldberg, and K. Deb. Long path problems. In Y. Davidor, H.-P. Schwefel, and R. Männer, editors, *Parallel Problem Solving From Nature – PPSN III, volume 866 of Lecture Notes in Computer Science*, pages 149–158, Berlin, 1994. Springer-Verlag.

[87] W. B. Provine. *Sewall Wright and Evolutionary Biology.* University of Chicago Press, Chicago, IL, 1986.

[88] D. B. Fogel and J. W. Atmar. Comparing genetic operators with gaussian mutations in simulated evolutionary processes using linear systems. *Biological Cybernetics*, 63:111–114, 1990.

[89] L. J. Eshelman and J. D. Schaffer. Crossover's niche. In S. Forrest, editor, *Genetic Algorithms: Proceedings of the Fifth International Conference (ICGA 1993)*, pages 9–14, San Mateo, CA, 1993. Morgan Kaufmann.

[90] W. M. Spears. Crossover or mutation? In L. D. Whitley, editor, *Foundations of Genetic Algorithms*, volume 2, pages 221–237, San Mateo, CA, 1993. Morgan Kaufmann.

[91] D. B. Fogel. Applying evolutionary programming to selected control problems. *Computers Math. Applic.*, 27(11):89–104, 1994.

[92] D. B. Fogel and L. C. Stayton. On the effectiveness of crossover in simulated evolutionary optimization. *BioSystems*, 32:171–182, 1994.

[93] U.-M. O'Reilly and F. Oppacher. Program search with a hierarchical variable length representation: Genetic programming, simulated annealing and hill climbing. In Y. Davidor, H.-P. Schwefel, and R. Männer, editors, *Parallel Problem Solving From Nature – PPSN III, volume 866 of Lecture Notes in Computer Science*, pages 397–406, Berlin, 1994. Springer-Verlag.

[94] U.-M. O'Reilly and F. Oppacher. Hybridized crossover-based search techniques for program discovery. Technical Report 95–02–007, Santa Fe Institute, Santa Fe, NM, February 1995.

[95] L. J. Eshelman, R. A. Caruana, and J. D. Schaffer. Biases in the crossover landscape. In J. D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 10–19, San Mateo, CA, June 4–7 1989. Morgan Kaufmann.

[96] S. Forrest and M. Mitchell. Towards a stronger building-blocks hypothesis: Effects of relative building-block fitness on GA performance. In *Foundations of Genetic Algorithms*, volume 2, pages 109–126, Vail, Colorado, 1993. Morgan Kaufmann.

[97] E. Brindle. *Genetic Algorithms for Function Optimization*. PhD thesis, University of Alberta, Edmonton, Canada., 1981. Available as TR 81–2, Department of Computer Science, University of Alberta.

[98] L. D. Whitley. The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In J. D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 116–121, San Mateo, CA, June 4–7 1989. Morgan Kaufmann.

[99] D. E. Goldberg and K. Deb. A comparative analysis of selection schemes used in genetic algorithms. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, volume 1, pages 69–93, San Mateo, CA, 1991. Morgan Kaufmann.

[100] K. A. De Jong. Genetic algorithms are NOT function optimizers. In L. D. Whitley, editor, *Foundations of Genetic Algorithms*, volume 2, pages 5–17, San Mateo, CA, 1993. Morgan Kaufmann.

[101] L. D. Whitley. Introduction. In L. D. Whitley, editor, *Foundations of Genetic Algorithms*, volume 2, pages 1–4, San Mateo, CA, 1993. Morgan Kaufmann.

[102] J. H. Holland. *Adaptation in Natural and Artificial Systems.* MIT Press, Cambridge, MA, 2nd edition, 1992.

[103] J. J. Grefenstette. Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics*, 16(1):122–128, Jan/Feb 1986.

[104] D. E. Goldberg. Simple genetic algorithms and the minimal, deceptive problem. In L. Davis, editor, *Genetic Algorithms and Simulated Annealing*, chapter 6, pages 74–88. Pitman (Morgan Kaufmann), London, 1987.

[105] K. Deb and D. E. Goldberg. Sufficient conditions for deceptive and easy binary functions. Technical report, University of Illinois, Urbana-Champaign, 1992. IlliGAL Report No 92001. Available via ftp from gal4.ge.uiuc.edu in pub/papers/IlliGALs/92001.ps.Z.

[106] T. Rado. On non-computable functions. *Bell System Technical Journal*, 41:877–884, 1962.

[107] A. H. Brady. The conjectured highest scoring machines for Rado's $\sigma(k)$ for the value $k = 4$. *IEEE Transactions on Electronic Computers*, EC-15:802, 1966.

[108] A. H. Brady. The determination of the value of Rado's noncomputable function $\sigma(k)$ for four-state Turing machines. *Mathematics of Computation*, 40(162):647–665, April 1983.

[109] H. Marxen and J. Buntrock. Attacking the busy beaver 5. *Bulletin of The European Association for Theoretical Computer Science*, 40:247–251, 1990.

[110] T. C. Jones and G. J. E. Rawlins. Reverse hillclimbing, genetic algorithms and the busy beaver problem. In S. Forrest, editor, *Genetic Algorithms: Proceedings of the Fifth International Conference (ICGA 1993)*, pages 70–75, San Mateo, CA, 1993. Morgan Kaufmann.

[111] R. G. Palmer, July 1992. Personal communication.

[112] J. H. Holland. Royal road functions. Internet Genetic Algorithms Digest v7n22. Available via ftp from ftp.santafe.edu in pub/terry/jhrr.tar.gz, Aug 12 1993.

[113] J. R. Levenick. Inserting introns improves genetic algorithm success rate: Taking a cue from biology. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 123–127, San Mateo, CA, 1991. Morgan Kaufmann.

[114] R. G. Palmer and C. M. Pond. Internal field distributions in model spin glasses. *Journal of Physics F*, 9(7):1451–1459, 1979.

[115] K. A. De Jong, W. M. Spears, and D. F. Gordon. Using Markov chains to analyze GAFOs. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms*, volume 3, San Mateo, CA, 1995. Morgan Kaufmann. (To appear).

[116] K. D. Boese, A. B. Kahng, and S. Muddu. A new adaptive multi-start technique for combinatorial global optimizations. *Operations Research Letters*, 16(2):101–113, September 1994.

[117] S. A. Kauffman. Adaptation on rugged fitness landscapes. In D. Stein, editor, *Lectures in the Sciences of Complexity*, volume 1, pages 527–618. Addison-Wesley Longman, 1989.

[118] S. A. Kauffman. *The Origins of Order; Self-Organization and Selection in Evolution.* Oxford University Press, New York, 1993.

[119] W. A. Tackett. *Recombination, Selection, and the Genetic Construction of Computer Programs.* PhD thesis, University of Southern California, Los Angeles, CA, April 1994.

[120] W. A. Tackett. Greedy recombination and genetic search on the space of computer programs. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms*, volume 3, San Mateo, CA, 1995. Morgan Kaufmann. (To appear).

[121] N. J. Nilsson and D. Rumelhart. Approaches to Artificial Intelligence. Technical Report 93–08–052, Santa Fe Institute, Santa Fe, NM, 1993. Summary of workshop held November 6–9, 1992. Available via ftp from ftp.santafe.edu in pub/Users/mm/approaches/approaches.ps.

[122] J.-L. Lauriere. A language and a program for stating and solving combinatorial problems. *Artificial Intelligence*, 10:29–127, 1978.

[123] S. Loyd. *Mathematical Puzzles of Sam Loyd.* Dover, new York, 1959.

[124] J. Slagle. A computer program for solving problems in freshman calculus (SAINT). *Journal of the Association for Computing Machinery*, 10(4):507–520, 1963. This also appears in E. A. Feigenbaum and J. Friedman (Eds.) *Computers and Thought* pp. 191–203. McGraw-Hill, New York, 1963.

[125] D. Partridge. *A New Guide to Artificial Intelligence*. Ablex Publishing Co., Norwood, NJ, 1991.

[126] J. L. Noyes. *Artificial Intelligence with Common Lisp*. D. C. Heath, Lexington, MA, 1992.

[127] P. H. Winston. *Artificial Intelligence*. Addison-Wesley, Reading, MA, 3 edition, 1992.

[128] G. F. Luger and A. Stubblefield. *Artificial Intelligence and the Design of Expert Systems*. Benjamin/Cummins, Redwood City CA, 2nd edition, 1993.

[129] W. B. Gevarter. *Artificial Intelligence, Expert Systems, Computer Vision and natural Language Processing*. Noyes Publications, Park Ridge, NJ, 1984.

[130] S. L. Tanimoto. *The Elements of Artificial Intelligence*. W. H. Freeman and Co., New York, 1990.

[131] D. W. Patterson. *Introduction to Artificial Intelligence and Expert Systems*. Prentice Hall, Englewood Cliffs, NJ, 1990.

[132] M. Ginsberg. *Essentials of Artificial Intelligence*. Morgan Kaufmann, San Mateo, CA, 1993.

[133] A. J. Gonzalez and D. D. Dankel. *The Engineering of Knowledge-Based Systems*. Prentice Hall, Englewood Cliffs, NJ, 1993.

[134] S. J. Scown. *The Artificial Intelligence Experience: An Introduction*. Digital Equipment Corporation, 1985.

[135] R. J. Schalkoff. *Artificial Intelligence: An Engineering Approach*. McGraw-Hill, New York, 1990.

[136] G. W. Ernst and A. Newell. *GPS:A case Study in Generality and Problem Solving*. Academic Press, New York, 1969.

[137] N. J. Nilsson. *Principles of Artificial Intelligence*. Tioga Publishing Co., Palo Alto, CA, 1980.

[138] I. Pohl. Bi-Directional search. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 6, pages 127–140, New York, 1971. American Elsevier.

[139] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[140] P. Helman. An algebra for search problems and their solutions. In L. Kanal and V. Kumar, editors, *Search in Artificial Intelligence*, pages 28–90. Springer Verlag, New York, 1988.

[141] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, pages 151–158, New York, 1971. Association for Computing Machinery.

[142] J. Doran and D. Michie. Experiments with the graph traverser program. *Proceedings of the Royal Society of London (A)*, 294:235–259, 1966.

[143] J. Horn and D. E. Goldberg. Genetic algorithm difficulty and the modality of fitness landscapes. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms*, volume 3, San Mateo, CA, 1995. Morgan Kaufmann. (To appear).

[144] A. D. Bethke. *Genetic Algorithms as Function Optimizers*. PhD thesis, University of Michigan, Ann Arbor, MI, 1981.

[145] L. D. Whitley. Fundamental principles of deception in genetic search. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, volume 1, pages 221–241, San Mateo, CA, 1991. Morgan Kaufmann.

[146] R. Das and L. D. Whitley. The only challenging problems are deceptive: Global search by solving order-1 hyperplanes. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 166–173, San Mateo, CA, 1991. Morgan Kaufmann.

[147] J. J. Grefenstette. Deception considered harmful. In L. D. Whitley, editor, *Foundations of Genetic Algorithms*, volume 2, pages 75–91, San Mateo, CA, 1993. Morgan Kaufmann.

[148] J. D. Schaffer, L. J. Eshelman, and D. Offutt. Spurious correlations and premature convergence in genetic algorithms. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, volume 1, pages 102–112, San Mateo, CA, 1991. Morgan Kaufmann.

[149] H. Kargupta and D. E. Goldberg. Decision making in genetic algorithms: A signal-to-noise perspective. Technical Report #94004, IlliGAL, University of Illinois, 1994.

[150] H. Kargupta. Signal-to-noise, crosstalk, and long range problem difficulty in genetic algorithms. In L. J. Eshelman, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, San Mateo, CA, 1995. Morgan Kaufmann. (to appear).

[151] Y. Davidor. Epistasis variance: A viewpoint on GA-hardness. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, volume 1, pages 23–35, San Mateo, CA, 1991. Morgan Kaufmann.

[152] R. Tanese. *Distributed Genetic Algorithms for Function Optimization.* PhD thesis, University of Michigan, Ann Arbor, MI, 1989.

[153] D. Lane, November 1994. Personal communication.

[154] D. E. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivation, analysis and first results. *Complex Systems*, 4:415–444, 1989.

[155] T. C. Jones. A description of Holland's royal road function. *Evolutionary Computation*, 2(4):411–417, 1995.

[156] G. Liepins and M. D. Vose. Deceptiveness and genetic algorithm dynamics. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, volume 1, pages 36–50, San Mateo, CA, 1991. Morgan Kaufmann.

[157] S. Forrest and M. Mitchell. What makes a problem hard for a genetic algorithm? Some anomalous results and their explanation. *Machine Learning*, 13:285–319, 1993.

[158] K. Deb and D. E. Goldberg. Analyzing deception in trap functions. In L. D. Whitley, editor, *Foundations of Genetic Algorithms*, volume 2, pages 93–108, San Mateo, CA, 1993. Morgan Kaufmann.

[159] R. A. Caruana and J. D. Schaffer. Representation and hidden bias: Gray vs. binary coding for genetic algorithms. In *Fifth International Conference on Machine Learning*, pages 153–161, Los Altos, CA, June 12–14 1988. Morgan Kaufmann.

[160] L. Davis. Bit climbing, representational bias and test suite design. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 18–23, San Mateo, CA, 1991. Morgan Kaufmann.

[161] S. Wright. Evolution in Mendelian populations. *Genetics*, 16:97–159, 1931.

[162] W. Fontana, P. F. Stadler, E. G. Bornberg-Bauer, T. Griesmacher, I. Hofacker, M. Tacker, P. Tarazona, E. D. Weinberger, and P. Schuster. RNA folding and combinatory landscapes. *Physical Review E*, 47(3):2083–2099, 1993.

[163] M. Huynen. *Evolutionary Dynamics and Pattern Generation in the Sequence and Secondary Structure of RNA: A Bioinformatic Approach*. PhD thesis, University of Utrecht, Netherlands, September 1993.

[164] E. D. Weinberger. Local properties of Kauffman's N-k model: A tunably rugged energy landscape. *Physical Review A*, 44(10):6399–6413, November 1991.

[165] K. E. Kinnear Jr. Fitness landscapes and difficulty in genetic programming. In *Proceedings of the First IEEE Conference on Evolutionary Computing*, pages 142–47, 1994.

[166] P. F. Stadler and W. Grüner. Anisotropy in fitness landscapes. *Journal of Theoretical Biology*, 165:373–388, 1993.

[167] P. Schuster and P. F. Stadler. Landscapes: Complex optimization problems and biopolymer structures. *Computers Chem.*, 18:295–314, 1994.

[168] P. F. Stadler. Linear operators on correlated landscapes. *J. Physique*, 4:681–696, 1994.

[169] T. E.. Davis and J. C. Principe. A simulated annealing like convergence theory for the simple genetic algorithm. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 174–181, San Mateo, CA, 1991. Morgan Kaufmann.

[170] M. D. Vose. Modeling simple genetic algorithms. In L. D. Whitley, editor, *Foundations of Genetic Algorithms*, volume 2, pages 63–73, San Mateo, CA, 1993. Morgan Kaufmann.

[171] J. Suzuki. A Markov chain analysis on a genetic algorithm. In S. Forrest, editor, *Genetic Algorithms: Proceedings of the Fifth International Conference (ICGA 1993)*, pages 146–153, San Mateo, CA, 1993. Morgan Kaufmann.

[172] G. Rudolph. Convergence analysis of conventional genetic algorithms. *IEEE Transactions on Neural Networks*, 5(1):96–101, Jan 1994. Special Issue on Evolutionary Computation.

[173] D. E. Goldberg and P. Segrest. Finite Markov chain analysis of genetic algorithms. In J. J. Grefenstette, editor, *Proceedings of the Second International Conference on Genetic Algorithms*, pages 1–8. Lawrence Erlbaum, Hillsdale, NJ, 1987.

[174] S. W. Mahfoud. Finite Markov chain models of an alternative selection strategy for the genetic algorithm. *Complex Systems*, 7(2):155–170, April 1993.

[175] J. Horn. Finite Markov chain analysis of genetic algorithms with niching. In S. Forrest, editor, *Genetic Algorithms: Proceedings of the Fifth International Conference (ICGA 1993)*, pages 110–117, San Mateo, CA, 1993. Morgan Kaufmann.

[176] L. D. Whitley and N.-W. Yoo. Modeling simple genetic algorithms for permutation problems. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms*, volume 3, San Mateo, CA, 1995. Morgan Kaufmann. (To appear).

[177] G. Wagner, January 1995. Personal communication.

[178] L. D. Whitley, July 1994. Personal communication.

[179] M. D. Vose, July 1994. Personal communication.

[180] L. Altenberg. The schema theorem and Price's theorem. In L. D. Whitley and M. D. Vose, editors, *Foundations of Genetic Algorithms*, volume 3, San Mateo, CA, 1995. Morgan Kaufmann. (To appear).

[181] B. T. Lowerre and R. D. Reddy. The Harpy speech understanding system. In W. A. Lea, editor, *Trends in Speech Recognition*. Prentice Hall, Englewood Cliffs, NJ, 1980.

[182] S. Lin and B. W. Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 31:498–516, 1973.

[183] K. D. Boese. Cost versus distance in the traveling salesman problem. Unpublished preliminary report., 1995.

[184] M. Padberg and G. Rinaldi. Optimization of a 532-city symmetric traveling salesman problem by branch and cut. *Operations Research Letters*, 6:1–7, 1987.

[185] L. Hagen and A. B. Kahng. Combining problem reduction and adaptive multi-start: A new technique for superior iterative partitioning. *IEEE Transactions on Computer Aided Design*, 1995. (to appear).

[186] W. Ruml, J. T. Ngo, J. Marks, and S. Shieber. Easily searched encodings for number partitioning. *Journal of Optimization Theory and Applications*, 1995. (to appear).

[187] N. Karmarkar and R. M. Karp. The differencing method of set partitioning. Technical Report UCB/CSD 82/113, University of California, Berkeley, Berkeley, CA, 1982.

[188] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, 1972.

[189] M. R. Garey and D. S. Johnson. *Computers and Intractibility: A Guide to the Theory of NP-Completeness.* W. H. Freeman, New York, 1979.

[190] P. Orponen, K.-I. Ko, U. Schöning, and O. Watanabe. Instance complexity. *Journal of the Association for Computing Machinery*, 41(1):96–121, January 1994.

[191] J. Doran. New developments of the graph traverser. In E. Dale and D. Michie, editors, *Machine Intelligence*, volume 2, pages 119–135, New York, 1967. American Elsevier.

[192] N. T. J. Bailey. *Statistical Methods in Biology.* Cambridge University Press, Cambridge, UK, 3rd edition, 1995.