Pokémon Cards and the Shortest Common Superstring

Mark Stamp Austin E Stamp

June 12, 2003

Abstract

Evidence is presented that certain sequences of Pokémon cards are determined by selecting consecutive elements from a longer sequence. We then consider the problem of recovering the shortest common superstring (SCS), i.e., the shortest string that contains each of the Pokémon card sequences as a consecutive substring. The SCS problem arises in many applications, most notably in DNA sequencing.

1 Introduction

Pokémon are mythical bug-like creatures invented by Satoshi Tajiri [1]. They were originally known as "Poketto Monstaa," Japanese for "Pocket Monster", which was eventually shortened to Pokémon.

Pokémon first appeared in 1996 as characters in a Game Boy game that was available only in Japan. This was followed by a Japanese cartoon, or anime, series[1]. The game and television series became world-wide successes.

In 1999 a highly successful Pokémon trading card game was introduced. The game is sometimes described as a sophisticated form of "rock, paper, scissors." In the Pokémon version, each trainer (i.e., player) has a set of trading cards that are used to battle an opposing trainer's cards. So-called booster packs of cards are available so that a trainer can create decks of cards with differing characteristics.

2 Booster packs

We consider 11-card Pokémon booster packs from the "base one, unlimited edition" series. In this series there are 102 distinct cards, consisting of 32 common cards, 6 energy cards, 32 uncommon cards, 16 rare cards and 16 rare holofoil cards. The 32 common cards are numbers 43 through 69 (Pokémon cards) and 91 through 95 (trainer cards), the energy cards are numbers 97 through 102, the uncommon cards are 23 through 42 (Pokémon) and 80 through 90 (trainer) and 96 (uncommon energy), the rare holofoil cards are numbered 1

[1]See [2] for an unusual story concerning the original Pokémon anime series.

through 16, and the rare non-holofoil cards are 17 through 22 (Pokémon) and 70 through 79 (trainer).

Each 11-card booster pack contains five common cards, two energy cards, one rare card, and three uncommon cards. Each type of card occurs consecutively as a block, but the order in which these blocks appear can vary from one printing to another. The article [3] has more details about Pokémon and the trading cards considered in this paper.

3 Shortest common superstring

Consider the following four sequences, or strings, of common Pokémon cards

$$\begin{aligned} A &= (95, 64, 44, 53, 57) \\ B &= (64, 44, 53, 57, 59) \\ C &= (44, 53, 57, 59, 91) \\ D &= (91, 56, 46, 45, 52). \end{aligned}$$

Observe that strings A and B overlap and the superstring $(95, 64, 44, 53, 57, 59)$ is the shortest string that contains both A and B.

For common Pokémon cards, we consider the problem of finding the shortest common superstring (SCS), i.e., the superstring of shortest length that contains all of the common card sequences as consecutive substrings. The SCS problem arises in many applications including DNA sequencing and data compression [4].

Let $o(X, Y)$ be the maximum overlap when string X is followed by string Y. To solve the SCS problem for the four common Pokémon card strings listed above, we first construct the complete directed graph G in Figure 1, where the directed edge between vertices X and Y is given the edge weight $o(X, Y)$. Then a solution to the SCS problem is a Hamiltonian path (not necessarily a circuit) in G of maximum total edge weight.

For the graph in Figure 1 the optimal solution is

$$A \to B \to C \to D$$

which has total edge weight 9 and consequently yields an SCS of length 11. The SCS is listed below with the initial point of each of the four strings indicated

$$\underset{A}{95}, \underset{B}{64}, \underset{C}{44}, 53, 57, 59, \underset{D}{91}, 56, 46, 45, 52. \tag{1}$$

As an aside, it is interesting to note that the traveling salesman problem (TSP) can be formulated in a manner very similar to the SCS problem. The significant differences between the two problems are that for the TSP the underlying graph is undirected and the goal is to find the Hamiltonian path of minimum total edge weight [5].

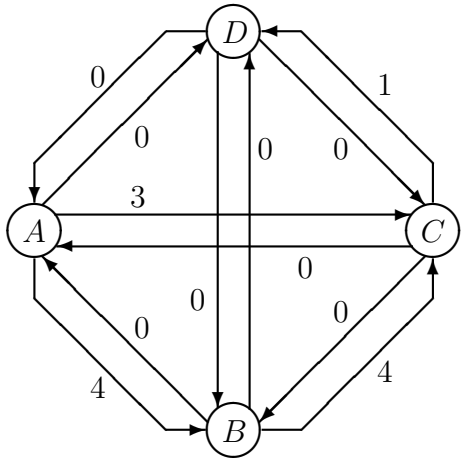

Figure 1: Directed graph G

4 Common card strings

We examined 153 booster packs, carefully recording the common card string that appears in each pack. The resulting 153 common card strings contain a large number of duplicates, leaving only 88 distinct strings. Three of these 88 appear to have a pair of elements transposed. We believe this is the case since the offending pairs do not appear within any other strings, but if swapped, each appears as a substring multiple times within other strings. Thus, we consider a set of 85 distinct strings, each of length five.

Recall that there are 32 distinct common Pokémon cards. In the 85 strings under consideration each of the 32 distinct cards is succeeded by at most four different cards. This observation led us to conjecture that the common cards are drawn from a superstring of length 128, with each of the 32 distinct cards appearing four times within the superstring. Below, we provide evidence that this is (approximately) the case.

Let N be the number of strings under consideration, L the length of the underlying superstring, n the number of distinct symbols and m the length of each string. We assume that all strings are of the same length. In the Pokémon problem we have $N = 85$, $n = 32$, $m = 5$ and L is to be determined. We assume that the strings are not allowed to "wrap", and hence, there are $L - m + 1$ positions at which the initial point of a string can occur.

Since $N < L - m + 1$, we can interpret

$$p = \frac{N}{L - m + 1} \tag{2}$$

as the probability of a sequence starting at a given point within the superstring and $1 - p$ is the probability that no string begins at a specified position. It follows that the expected number of strings that overlap in k positions is about

$$\frac{N^2}{L - m + 1} \tag{3}$$

and this holds for any $k \in \{1, 2, \ldots, m-1\}$. Since we do not allow duplicate sequences, $k = m$ cannot occur.

Let s_i be the ith Pokémon card string. Define $\mathcal{O}$ to be the 85×85 overlap matrix given by $\mathcal{O}_{ij} = o(s_i, s_j)$. For the 85 sequences under consideration, the values in $\mathcal{O}$ are summarized in Table 1, where we have ignored the diagonal elements.

overlap	number
0	6744
1	225
2	54
3	56
4	61

Table 1: Summary of overlap matrix $\mathcal{O}$

Suppose that the Pokémon common card strings are selected at random (with replacement, for simplicity) and each symbol is equally likely. Then the expected number of strings with overlap of length two is approximately

$$\frac{N(N-1)}{n^2},$$

which, for $N = 85$ and $n = 32$, is about 7. A similar calculation shows that the expected number of strings with overlap of length three or four are both less than one.

We would therefore expect at random to find about 7 pairs with an overlap of length two and no such random matches for any length greater than two. In Table 1 we see that the number of pairs with overlap of length two is actually less than the number with overlap three or four. One explanation for this is that every overlap of length causal (i.e., drawn from the superstring), not a random match. This could occur if the superstring was chosen so that no consecutive pair of elements is repeated within the superstring. We briefly return to this point in the next section.

Ignoring any random overlap of length two, we average the number of overlaps of length 2,3 and 4 from Table 1 and obtain the value 57. We then set equation (3) equal to 57 and solve for the unknown value L, to find $L = 130.75$. This result is close to our conjectured value of $L = 128$.

These calculations indicate that the Pokémon card strings are not randomly selected. Also, the numbers are consistent with our conjecture that the strings are selected from a superstring of about 128 elements.

As mentioned above, any overlap of length three or four is, with high probability, causal. However, an overlap of length three, for example, might not contribute any new information. In particular, if an overlap of length three coincides with two pairs of strings that have overlap of length four, then it only serves to reinforces information that is already known to be almost certainly correct. For example, consider the three strings A, B and C from Figure 1. We

have $o(A, B) = o(B, C) = 4$ and $o(A, C) = 3$. However, the overlap of length three does not contribute any useful information. Of course, a similar situation can arise with strings that overlap in fewer than three element. This motivates the following definition.

Definition 1 *We refer to pairs of strings that have the maximum overlap within a block as* relevant*, while those that do not are* irrelevant*.*

In Figure 1, the pairs (A, B) and (B, C) are relevant, while (A, C) is irrelevant.

The expected number of relevant pairs of strings can be estimated. For example, a pair of strings with overlap of length three is relevant provided that no string begins at the one superstring position that lies between the starting points of the pair of strings. The probability of this is $1 - p$, where p is given in (2).

In the Pokémon problem, all pairs of strings with overlap of length three are almost certainly causal, but only about $56(1-85/124)$ or about 18 of these will be relevant. Similarly, about 6 of the overlaps of length two and about 2 of the overlaps of length one are relevant. In addition, we expect less than one of the overlaps of length zero to be relevant. To have any hope of recovering the true SCS, we must have no relevant overlap of length zero—otherwise we would need to select matching pairs of strings where no score is available.

In general, about

$$\frac{N^2}{L-m+1}\left(1-\frac{N}{L-m+1}\right)^{m-k-1} \tag{4}$$

of the causal k-overlapping pairs will be relevant. To simplify the recovery of the Pokémon SCS, it would be useful to have no relevant overlaps of length one. In light of (4), and assuming $L = 128$, we could attempt to increase N to the point where

$$\frac{N^2}{124}\left(1-\frac{N}{124}\right)^3 < 1$$

We find that any $N \geq 95$ will suffice. Similarly, we would need $N \geq 112$ before the expected number of relevant overlaps of length two would fall below one. With the 85 available Pokémon strings, we expect to be able to construct a shortest common superstring for the the Pokémon problem using only overlaps of length one and greater. In the next section we make use of this observation to devise a simple greedy algorithm to solve the Pokémon SCS problem.

5 A greedy solution

Suppose we have N strings, say, s_0 through s_{N-1}. Then one approach to finding the exact SCS solution would be to compute the overlap score associated with each of the $N!$ permutations of $\{s_0, s_1, s_2, \ldots, s_{N-1}\}$. A permutation with maximum score would yield a solution to the SCS problem. Of course, the work factor for this approach is prohibitive, even for relatively small values of N.

As a decision problem, the shortest common superstring is known to be NP-complete [6]. Hence, in general, we cannot expect to find the exact SCS solution efficiently. However, there

are efficient techniques that are known to give good approximate solutions. For example, dynamic programming or a greedy algorithm will produce a solution whose length is within a small multiple of the SCS [4, 7, 8].

For the Pokémon SCS problem the 85×85 overlap matrix $\mathcal{O}$ is sparse, containing only about 400 nonzero entries. Furthermore, Table 1 indicates that all pairs with overlap two or greater are likely causal. As discussed in the previous section, we also expect all but about two of the relevant overlaps to be of length two or greater.

Given these observations, we implemented a simple greedy approach that includes branching in the case of all tie scores meeting or exceeding a given threshold The algorithm begins by initiating a permutation for each of the N strings,

$$\text{perm}[i][0] = i, \text{ for } i = 0, 1, \ldots, N-1.$$

Then we proceed to greedily construct a complete permutation out of each partial permutation by appending the eligible string with maximum overlap at each step. However, if there is more than one candidate string that attains the maximum overlap at a given step, and that overlap meets or exceeds a bound k, we create a new partial permutation (or branch) for each such string.

From the definition of relevant strings, it is clear that this greedy algorithm will yield all possible shortest common superstrings, provided k is chosen so that all relevant overlaps are of length k or greater. We state this result as a theorem.

Theorem 1 *A greedy algorithm with branching on all ties of overlap k or greater will yield a shortest common superstring (SCS) for the given data set provided there is no relevant overlap of length less than k.*

The work factor for this greedy branching algorithm depends on the number of ties that meet or exceed the bound k. Generating and scoring any one permutation of length N requires on the order of N^2 work. The total work is therefore bounded by MN^2, where M is the final number of permutations (i.e., superstrings). If we have chosen k correctly, then the highest scoring of these M superstrings will necessarily yield a shortest common superstring. We can obtain an estimate for k using the methods of the previous section.

Initially, there are N partial permutations. With p as given in (2), the probability that any (partial) permutation has a relevant overlap of length k is

$$p_k = p(1-p)^{m-k-1} \text{ for } k \in \{1, 2, \ldots, m-1\}$$

and

$$p_0 = 1 - \sum_{i=1}^{m-1} p_i.$$

Also, for any given partial permutation of length i, the expected number of random length one overlaps is $(N-i)/n$. Combining these observations, we see that the expected value of M can be approximated by the product

$$M = N \prod_{i=1}^{N-1} \left(1 + x\, \frac{N-i}{n}\right),$$

where x is the probability of an overlap (causal or random) of length k.

For the Pokémon problem, we set $x = p_0 + p_1$, since we expect random overlaps of length one whenever the causal overlap is of length zero or one. For $N = 85$, $m = 5$, $n = 32$, $k = 1$ and a range of values of L, the computed expected values of M are plotted in Figure 2. This graph indicates that if we are correct in our estimates of the length of the SCS for the Pokémon sequences, then the amount of branching required to solve the problem will be easily manageable.

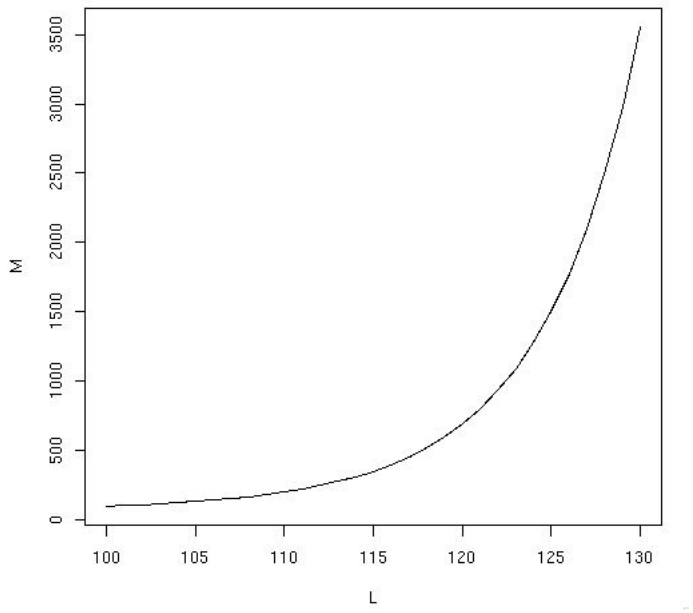

Figure 2: $E(M)$ for $N = 85$, $m = 5$ and $n = 32$

For the Pokémon problem, we begin with $N = 85$ branches (i.e., partial permutations) and we set $k = 1$. The greedy-branching algorithm finishes with 403 superstrings (i.e., complete permutations). The best scoring permutation has an overlap of 303 and hence yields a common superstring of length 122. If our assumption that all relevant overlaps are of length $k = 1$ or greater is correct, then we have found a shortest common superstring.

The superstring found by the algorithm is listed below, with the initial position of each the 85 strings (reordered so they are in sequence) listed beneath the corresponding super-

string element.

$$
\begin{array}{cccccccccccccccc}
93 & 68 & 51 & 46 & 62 & 95 & 64 & 44 & 53 & 57 & 59 & 91 & 56 & 46 & 45 & 52 \\
0 & 1 & 2 & & & 3 & 4 & 5 & & & 6 & 7 & 8 & & & 9 \\
49 & 92 & 66 & 65 & 63 & 94 & 50 & 55 & 68 & 48 & 43 & 67 & 69 & 62 & 51 & 54 \\
10 & 11 & & & 12 & 13 & 14 & 15 & & 16 & 17 & & 18 & 19 & 20 & 21 \\
58 & 47 & 61 & 60 & 64 & 65 & 67 & 68 & 57 & 44 & 62 & 59 & 69 & 52 & 66 & 58 \\
 & 22 & & 23 & 24 & & & & 25 & 26 & & 27 & 28 & 29 & 30 & 31 \\
49 & 93 & 43 & 56 & 60 & 53 & 63 & 55 & 45 & 95 & 61 & 92 & 48 & 54 & 50 & 46 \\
32 & & 33 & 34 & 35 & 36 & 37 & & 38 & & 39 & & 40 & 41 & 42 & 43 \\
47 & 51 & 64 & 59 & 68 & 62 & 57 & 91 & 67 & 51 & 60 & 52 & 65 & 54 & 55 & 95 \\
44 & 45 & 46 & 47 & 48 & 49 & & 50 & 51 & & 52 & & 53 & 54 & 55 & 56 \\
47 & 53 & 45 & 93 & 44 & 63 & 49 & 43 & 50 & 56 & 61 & 58 & 94 & 48 & 69 & 46 \\
 & 57 & 58 & 59 & 60 & & 61 & 62 & 63 & 64 & 65 & 66 & 67 & 68 & & 69 \\
66 & 64 & 50 & 45 & 58 & 59 & 92 & 43 & 52 & 57 & 55 & 49 & 94 & 44 & 56 & 65 \\
70 & 71 & 72 & 73 & 74 & 75 & 76 & & & 77 & 78 & 79 & & & 80 & 81 \\
48 & 53 & 91 & 61 & 63 & 54 & 47 & 60 & 93 & 66 & & & & & & \\
82 & & & & 83 & 84 & & & & & & & & & &
\end{array}
\tag{5}
$$

Each Pokémon common card number occurs four times in the superstring (5), except for 67,69,91,92,94 and 95, each of which appears three times. Consequently, the length of the superstring is slightly less than conjectured. It is possible that this is simply due to a lack of data at either or both ends of the superstring. It is also the case that every overlap of length two or greater is causal. Apparently, the superstring was carefully chose so as to avoid any repeated 2-element patterns. In addition, any five-element string selected from the superstring (5) does not contain duplicate card numbers. This feature insures distinct common cards within a Pokémon card booster pack.

6 Data

The complete set of data for the Pokémon cards analyzed in this paper is available at

`http://home.earthlink.net/~mstamp1/pokemon/pokemondata.html`

More data will be added as time and budget permits.

References

[1] Pokémon: A quick history, at `http://members.tripod.com/animefan25/pkmnhistor.htm`

[2] The Pokémon seizure incident, at `http://members.tripod.com/animefan25/seizures.htm`

[3] M. Stamp and A. Stamp, Pokémon trading card sequences, at `http://home.earthlink.net/~mstamp1/papers/pokemon.pdf`

[4] A. Frieze and W. Szpankowski, Greedy algorithms for the shortest common superstring that are asymptotically optimal, *Algorithmica*, Vol. 21, 1998, pp. 21-36.

[5] D.L. Kreher and D.R. Stinson, *Combinatorial Algorithms: Generation, Enumeration and Search*, CRC Press, 1999.

[6] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman and Company, 1979, p. 228.

[7] A.G. Percus and D.C. Torney, Greedy algorithms for optimized DNA sequencing, Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 1999), pp. 955-956.

[8] J.S. Turner, Approximation algorithms for the shortest common superstring problem, *Information and Computation*, Vol. 83, No. 1, pp. 21-40, October 1989.