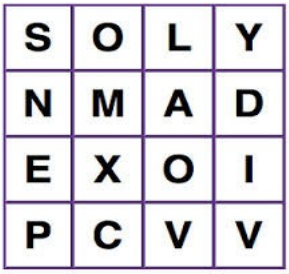
High-Scoring Boggle Board Generation Using Simulated Annealing and Genetic Algorithms

Theresa Guinard

Problem Description

The objective of this project is to generate a Boggle board with as high of a score as possible. A Boggle board is a four-by-four grid of English letters. Any square is allowed to be any English letter. An example Boggle board is shown below:



A word is formed on a Boggle board as a traversal of adjacent squares (any square that shares a corner or edge with the original square). The traversal may begin on any square, and each square may be visited and used up to once while forming a word. In the above board, “pen” and “mold” can be formed, but not “pep”, as the letter “p” would have to be visited twice.

The score of a Boggle board is the sum of the scores of all distinct English words that can be formed in the board. The score for a word is based on its length, as defined in the table below:

|word|	**Score**
3,4	1
5	2
6	3
7	5
8+	11

The objective of this project is to generate a Boggle board with as high of a score as possible. A valid list of English words is predefined by a dictionary text file.

Methodology

No efficient, exact solution is known for this problem, and it is possible that no such solution exists. A brute force search must examine 26^{16} possible solutions, which is highly impractical on any modern hardware. Therefore search heuristics will be used. Two separate techniques were investigated: genetic algorithms and simulated annealing.

Genetic Algorithms

Genetic algorithms keep track of an evolving set of possible solutions, a "population." Each iteration, the best members of the population combine to form a the next generation, and mutations and crossovers can occur in the process. Optionally, some of the best boards survive to the next generation. This is called the elite population.

The Boggle score of each board was used to find the best members of the population for reproduction and elite survival. To combine two boards, for each position in the grid, a random parent board was chosen, and that parent's corresponding letter in the given position was placed in the given position of the new board. Mutations were defined as replacing a random letter on a random board position. Crossovers were defined as transposing two adjacent letters.

In this project, a population size of 100 was used. The 30 highest-scoring boards were allowed to reproduce each generation, and these boards were randomly paired. In addition, the 10 highest-scoring boards made up the elite population that survived to the next generation. Mutations and crossovers occurred upon the creation of a new board after mating. Multiple crossovers and mutations were allowed, and there was a 0.2^n chance of at least n mutations, and a 0.2^n chance of at least n crossovers for a given board. These parameters were also modified for various experiments. Each experiment was run for 1000 generations.

Simulated Annealing

Simulated annealing starts with a random candidate solution, then modifies the solution over a series of iterations. Each modification to the solution can be accepted or rejected. A change that improves the score of the solution is automatically accepted, and a change that lowers the score is accepted with a certain probability.

The temperature is a function of the current iteration:

$$T = \text{maxTemp} * (1 - \text{iterations} / \text{maxIterations})$$

The probability of accepting a solution is a function of the temperature and the change in score. The following is a standard acceptance probability function used in simulated annealing, and was used in this project [4]:

$$p = \min(1, e^{(\text{newScore} - \text{score})/T})$$

All simulated annealing experiments in this project used 1000 iterations. Three changes were applied to the board each iteration, and each change was evaluated separately for whether it would be accepted. The first change was identical to mutation in the genetic algorithm, and the second change was identical to crossovers in the genetic algorithm. The third change swapped two random letters in the board. A variety of maximum temperatures were used.

Dataset

The only data used from an outside source was the dictionary file; all boards were generated internally. The dictionary used for this project was the YAWL word list found at [1]. This was also the word list used in [2] and [3], two other recreational attempts to solve this problem.

Results

Optimal Random Letter Generation Technique

Two different ways of generating random letters were tested. The first method involved selecting each letter with equal probability, which is called “uniform” in the chart below. The second method involved selecting each letter with probability proportional to its frequency in a dictionary. This method is called “dictionary” in the chart below.

Random letters are generated on two occasions (for both algorithms): first when generating random initial boards, and then when generating random letters for mutations.

All combinations of random letter generating techniques were tried. Each score in the table below is the average over four trials.

	Uniform initial Uniform mutation	**Uniform initial Dictionary mutation**	**Dictionary Initial Uniform mutation**	**Dictionary Initial Dictionary mutation**
Simulated Annealing	3783.5	4073.25	3464.75	3918.75
Genetic Algorithms	3377.75	3826.5	3725	3679.5

For both algorithms, starting with uniform initial boards and having dictionary mutations yielded optimal results. This technique was used to produce the rest of the results in this paper.

Genetic Algorithms: Optimal Setup

The original setup of the genetic algorithm consisted of the following constraints:

- Population size of 100
- Reproducing population of 30
- Elite population of 10
- Only one copy of a given board is allowed to exist
- 0.2^n chance of n or more crossovers
- 0.2^n chance of n or more mutations

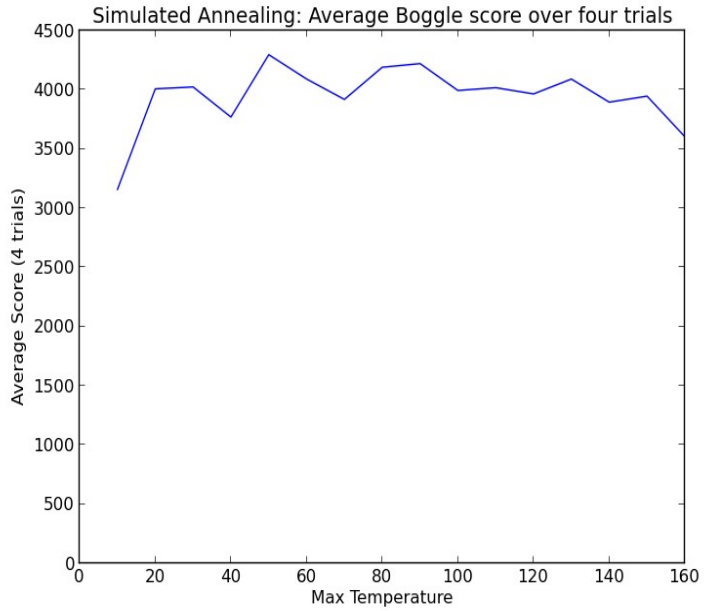
The original setup described above was used, then various parameters were modified. Each setup was run for two trials.

Setup	**Trial 1 Score**	**Trial 2 Score**
Original	3604	3575
No Elite Population	3042	3481
Multiple Copies of Boards Allowed	4236	4540
No Crossovers	3166	2717
No Mutations	2499	3675

The only modification that clearly improved results was allowing for multiple copies of boards. This modified setup was selected as the optimal algorithm for the genetic algorithms approach.

Simulated Annealing: Comparing Temperatures

Simulated annealing was run using various maximum temperatures. The results are not greatly affected by the maximum temperature, but a maximum temperature between about 40 and 120 seems to be optimal. A maximum temperature of 70 was selected as the optimal algorithm for the simulated annealing approach.



Comparison of Methods

The optimal setups found for genetic algorithms and simulated annealing were each run for 8 trials. Both methods performed comparably, but genetic algorithms obtained slightly higher scores, and simulated annealing was more efficient in its run time. In terms of the scores of the generated boards, both algorithms clearly perform better than random board generation (both generated with dictionary-frequency probability and uniform probability for letter selection).

Technique	**Average Score (over 8 trials)**	**High Score (over 8 trials)**	**Run time**
Genetic Algorithms (multiple board copies allowed)	4117.5	4540	53 minutes
Simulated Annealing (maxTemp = 70)	4007.6	4336	3 minutes
Random Board (dictionary probability)	270.75	419	0.8 seconds
Random Board (uniform probability)	51.25	151	0.3 seconds

Highest-scoring Board Found

The highest-scoring board found was with genetic algorithms, and was 4540 points:

B	L	P	S
E	A	I	E
S	T	R	N
T	E	S	G

This board contains 1351 words, and the three longest words are “splattering,” “plasterings,” and “terneplates.” This is higher than two other attempts at this problem, which found boards with scores of 4527 [2] and 4410 [3] (both attempts use the same YAWL dictionary as used in this project).

Conclusion

Simulated annealing and genetic algorithms were both successful approaches to finding high-scoring Boggle boards. The genetic algorithm was able to find a higher scoring board than other approaches to this problem. More work could be done to optimize the performance of genetic algorithms, and both algorithms could be further experimented with to find more optimal solutions.

References

[1] "Yet Another Word List [YAWL]." Computer Science Education at Duke. http://www.cs.duke.edu/csed/data/yawl-0.3.2/

[2] Do, Chuong (Tom). "Creating Dense Boggle Boards." http://ai.stanford.edu/~chuongdo/boggle/

[3] Dave, Ankur. "Optimizing Boggle Boards: An Evaluation of Parallelizable Techniques." http://ankurdave.com/dl/AnkurDaveExtendedEssay2009.pdf

[4] Henderson, Darrall, Sheldon H. Jacobson, and Alan W. Johnson. *The Theory and Practice of Simulated Annealing*. http://homes.ieu.edu.tr/~agokce/Courses/Chapter%208%20Theory%20and%20Practice%20of%20simulated%20Annealing.pdf