# A Genetic Algorithm with Best Combination Operator for the Traveling Salesman Problem

Muhammad L. Shahab, Titin J. Ambarwati, Soetrisno and Mohammad I. Irawan

***Abstract*—In this research, we propose a genetic algorithm with best combination operator ($BC_{x,y}O$) for the traveling salesman problem. The idea of best combination operator is to find the best combination of some disjoint sub-solutions (also the reverse of sub-solutions) from some known solutions. We use $BC_{2,1}O$ together with a genetic algorithm. The proposed genetic algorithm uses the swap mutation operator and elitism replacement with filtration for faster computational time. We compare the performances of GA (genetic algorithm without $BC_{2,1}O$), $IABC_{2,1}O$ (iterative approach of $BC_{2,1}O$), and $GABC_{2,1}O$ (genetic algorithm with $BC_{2,1}O$). We have tested GA, $IABC_{2,1}O$, and $GABC_{2,1}O$ three times and pick the best solution on 50 problems from TSPLIB. From those 50 problems, the average of the accuracy from GA, $IABC_{2,1}O$, and $GABC_{2,1}O$ are 65.12%, 94.21%, and 99.82% respectively.**

***Index Terms*—Traveling salesman problem, genetic algorithm, operator, best combination.**

## I. Introduction

THE traveling salesman problem is a famous combinatorial problem which has been studied by many researchers. TSP has many applications in vehicle routing problem [1], transport routes optimization [2], air logistics [3], chemical shipping [4], bioinformatics [5], and many others.

There are a lot of methods that had been developed to solve the TSP. The easiest one is the nearest neighbor algorithm (always choose the next closest node). The nearest neighbor algorithm usually produce a sub-optimal route (except in trivial cases). Dynamic programming algorithm can find an optimal solution for small TSP. The idea is that in an optimal solution, the path through the remaining subset must be optimal [6], [7]. Lin-Kernighan heuristic algorithm makes a great improvement in the quality of solutions provided by another heuristic methods [6], [8]. Heuristic algorithms are often used because they are able to provide solutions in a faster time [9].

Population-based algorithms, such as genetic algorithms [10], are also widely used today. These algorithms can obtain a better solution than heuristic algorithms. Usually these algorithms use certain operators to get new solutions from existing solutions.

In this research, we propose a genetic algorithm with best combination operator ($BC_{x,y}O$) for the traveling salesman problem.

Manuscript received June 29, 2019; accepted August 31, 2019.

The authors are with the Department of Mathematics, Institut Teknologi Sepuluh Nopember, Surabaya 60111, Indonesia. E-mail: luthfishahab@matematika.its.ac.id

This research was supported by Institution of Research and Community Service (LPPM), Institut Teknologi Sepuluh Nopember (ITS), Ministry of Research, Technology, and Higher Education (KEMENRISTEKDIKTI). In accordance with the funding agreement of Penelitian Pemula, contract number: 1193/PKS/ITS/2019.

## II. The Traveling Salesman Problem

Suppose there are some nodes that are labeled by $1,2,\ldots,n$ and $d_{i,j}$ represents the distance from node $i$ to node $j$. In general, the distance can be obtained from traveling time, traveling distance, traveling cost, Euclidean distance, or other relations. The objective of the traveling salesman problem (TSP) is to find the shortest route that visits each node exactly ones and returns to the origin city.

A solution of a TSP can be written as a permutation $p_1 p_2 \ldots p_n$ of the elements $1,2,\ldots,n$. The distance of $p_1 p_2 \ldots p_n$ is calculated by

$$\sum_{i=1}^{n} d_{p_i,p_{i+1}} \tag{1}$$

where $p_{n+1} = p_1$ and $d_{p_i,p_{i+1}}$ is the distance from node $p_i$ to node $p_{i+1}$.

In this research, we focus on symmetric TSP, i.e. the distance from node $i$ to node $j$ is equal to the distance from node $j$ to node $i$.

### A. TSPLIB

Gerhard Reinelt published the TSPLIB in 1991 [11]. It is a collection of benchmark instances of varying difficulty, which has been used by many research groups for comparing results.

For a TSP with EUC_2D type, $d_{ij}$ is calculated by

$$d_{ij} = \left\lfloor \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} + 0.5 \right\rfloor \tag{2}$$

where $\lfloor x \rfloor$ is floor function.

The remaining types of TSP in TSPLIB and how to calculate the distances, can be read in [11], [9].

## III. Best Combination Operator

The idea of best combination operator is to find the best combination of some disjoint sub-solutions (also the reverse of sub-solutions) from some known solutions. We introduce an abbreviation $BC_{x,y}O$, where $x \geq 2$ and $y \geq 1$, to represent the best combination of $x$ disjoint sub-solutions from $y$ known solutions. It is the general form of best combination operator. The simplest one is $BC_{2,1}O$.

### A. Example of $BC_{2,1}O$

Suppose that there is a TSP consisting of $n = 6$ nodes, and the node coordinates are shown in TABLE I and Fig. 1.

TABLE I: The coordinates of the nodes

| $i$ | $x_i$ | $y_i$ |
|---|---|---|
| 1 | 0 | 10 |
| 2 | 10 | 10 |
| 3 | 20 | 10 |
| 4 | 20 | 0 |
| 5 | 10 | 0 |
| 6 | 0 | 0 |


Fig. 1: The coordinates of the nodes

From those coordinates, we can use (2) to obtain the following distance matrix

$$D = \begin{pmatrix} 0 & 10 & 20 & 22 & 14 & 10 \\ 10 & 0 & 10 & 14 & 10 & 14 \\ 20 & 10 & 0 & 10 & 14 & 22 \\ 22 & 14 & 10 & 0 & 10 & 20 \\ 14 & 10 & 14 & 10 & 0 & 10 \\ 10 & 14 & 22 & 20 & 10 & 0 \end{pmatrix} \quad (3)$$

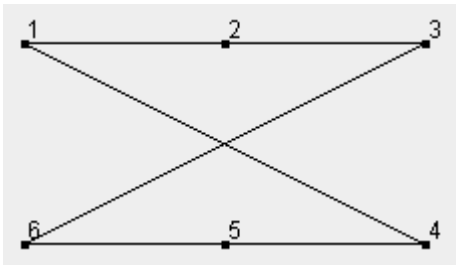


Fig. 2: Example of TSP solution

Suppose that 123654 is a random solution for the TSP. Using (1), the distance of this solution is equal to 84. First, we take all sub-solutions of length 2, 3, or 4 from 123654 as shown in Table II. The reverse of those sub-solutions are shown in Table III.

TABLE II: Sub-solutions of 123654

| Sub-solution of | | |
|---|---|---|
| length 2 | length 3 | length 4 |
| 12 | 123 | 1236 |
| 23 | 236 | 2365 |
| 36 | 365 | 3654 |
| 65 | 654 | 6541 |
| 54 | 541 | 5412 |
| 41 | 412 | 4123 |

To obtain different solutions from the initial solution, we search pairs of two disjoint sub-solutions from Table II and Table III.

TABLE III: The reverse of sub-solutions of 123654

| The reverse of sub-solution of | | |
|---|---|---|
| length 2 | length 3 | length 4 |
| 21 | 321 | 6321 |
| 32 | 632 | 5632 |
| 63 | 563 | 4563 |
| 56 | 456 | 1456 |
| 45 | 145 | 2145 |
| 14 | 214 | 3214 |

TABLE IV: Pairs of two disjoint sub-solutions and the new solutions

| Sub-solution from | | New Solution |
|---|---|---|
| Table II | Table III | |
| 12 | 4563 | 124563 |
| 23 | 1456 | 231456 |
| 36 | 2145 | 362145 |
| 65 | 3214 | 653214 |
| 54 | 6321 | 546321 |
| 41 | 5632 | 415632 |
| 123 | 456 | 123456 |
| 236 | 145 | 236145 |
| 365 | 214 | 365214 |
| 654 | 321 | 654321 |
| 541 | 632 | 541632 |
| 412 | 563 | 412563 |
| 1236 | 45 | 123645 |
| 2365 | 14 | 236514 |
| 3654 | 21 | 365421 |
| 6541 | 32 | 654132 |
| 5412 | 63 | 541263 |
| 4123 | 56 | 412356 |

Since we focus on symmetric TSP, there is some equal new solutions in Table IV, i.e. 124563 is equal to 365421, 123456 is equal to 654321, and so on. If we remove unnecessary solutions, and then count the distance of the remaining solutions, we will get results as shown in Table V. Because the best solution is 123456, we pick it as a new solution. Its distance is equal to 60.

This is an example of $\mathrm{BC}_{2,1}\mathrm{O}$. We can use $\mathrm{BC}_{x,y}\mathrm{O}$ for two or more solutions using similar steps as before.

### *B. $BC_{2,1}O$ Simplification*

We can make a simplification for $\mathrm{BC}_{2,1}\mathrm{O}$. The purpose of the simplification is to reduce the computational time. We write again the solutions listed in Table V. It is easy to see the difference between the initial solution and new solutions in Table VI. The second and third column have same solutions, we just change the starting node and the direction.

The first six new solutions are obtained by reversing a sub-solutions of length 2. The remaining three new solutions are obtained by reversing sub-solutions of length 3.

From Table IV, V, and VI, it can be seen that we will get all of the different new solutions by reversing a sub-solutions of length 2 and 3. If we have a TSP with $n$ nodes, then we need to reverse sub-solutions of length 2, 3, ..., and $n/2$.

Suppose that there is a solution $p_1 p_2 \dots p_n$ and its distance is $x$. If we reverse the order of $p_i p_{i+1} \dots p_{j-1} p_j$, in $p_1 p_2 \dots p_n$, we will get

TABLE V: The new solutions

| Solution | Distance |
|---|---|
| 124563 | 86 |
| 231456 | 86 |
| 362145 | 92 |
| 653214 | 86 |
| 546321 | 86 |
| 415632 | 92 |
| 123456 | 60 |
| 236145 | 84 |
| 365214 | 84 |

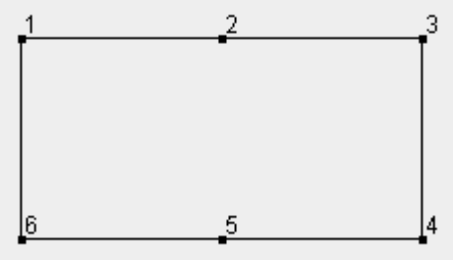


Fig. 3: New solution obtained by $BC_{2,1}O$

$$p_1 p_2 \dots p_{i-1} p_j p_{j-1} \dots p_{i+1} p_i p_{j+1} \dots p_n$$

and its distance is equal to

$$x - d_{p_{i-1},p_i} - d_{p_j,p_{j+1}} + d_{p_{i-1},p_j} + d_{p_i,p_{j+1}} \quad (4)$$

Using this simplification, the objective of $BC_{2,1}O$ is to find the best $i$ and $j$ so that the value obtained by (4) is as small as possible. The pseudocode of $BC_{2,1}O$ can be seen in Algorithm 1.

### C. Iterative Approach of $BC_{2,1}O$

After we get a new solution from $BC_{2,1}O$, we can apply the same process again to the new solution. That operator can be used iteratively until there is no further improvement. The initial solution can be any random permutation. Usually, we will get different final solutions if the initial solutions are not equal.

### D. Proposed Genetic Algorithm

It is not enough to solve the traveling salesman problem only using $BC_{2,1}O$ or the iterative approach, so we use the help of a genetic algorithm. The proposed genetic algorithm uses the swap mutation operator and elitism replacement with filtration for faster computational time.

Suppose that there is a solution $p_1 p_2 \dots p_n$ and random different values $i$ and $j$, where $i, j \in \{1, 2, \dots, n\}$. The swap mutation is done by swapping the position of $p_i$ and $p_j$, i.e. if the initial solution is $p_1 p_2 \dots p_{i-1} p_i p_{i+1} \dots p_{j-1} p_j p_{j+1} \dots p_n$, then the new solution is $p_1 p_2 \dots p_{i-1} p_j p_{i+1} \dots p_{j-1} p_i p_{j+1} \dots p_n$. In this research, every solution in the population is mutated to produce a new solution.

To get $N$ solutions for a new population, where $N$ is the size of the population, we use elitism replacement with filtration. First, we put together $N$ solutions from the initial population and $N$ new solutions obtained by swap mutation. If there are two identical solutions in the population, we pick one of them and remove the other one. With these steps, it can be guaranteed that all solutions are different. Then, we sort them according to their distance. And then we pick $N$ best solutions for the new population.

There are two stopping conditions used in this research. The first one, GA will stop if he has found the optimal solution. We can use this stopping condition because of the optimal solution of every problem in TSPLIB is known. The second one, GA will stop if the maximum computational time is reached. The maximum computational time used in this research is 100 seconds.

You can access the source code used in this research freely on https://github.com/mlshahab/gabcotsp.

TABLE VI: The initial and new solutions

| Initial Solution | New Solution | |
|---|---|---|
| 123654 | 124563 | 213654 |
| | 231456 | 132654 |
| | 362145 | 126354 |
| | 653214 | 123564 |
| | 546321 | 123645 |
| | 415632 | 423651 |
| | 123456 | 123456 |
| | 236145 | 523614 |
| | 365214 | 143652 |

**input :** $n$ (the number of nodes),
$p_1 p_2 \dots p_n$ (a solution)
**output:** $p_1 p_2 \dots p_n$ (new solution)

**for** $i \leftarrow 2$ **to** $n/2$ **do**
  **for** $j \leftarrow 1$ **to** $n$ **do**
    $a \leftarrow (j-1+n) \bmod n$ ;
    $b \leftarrow j$ ;
    $c \leftarrow (j+i-1) \bmod n$ ;
    $d \leftarrow (j+i) \bmod n$ ;
    $e \leftarrow p_a$ ;
    $f \leftarrow p_b$ ;
    $g \leftarrow p_c$ ;
    $h \leftarrow p_d$ ;
    $s_{i,j} \leftarrow -d_{e,f} - d_{g,h} + d_{e,g} + d_{f,h}$ ;
  **end**
**end**
$(i, j) \leftarrow$ IndexOfMinimumElement$(s)$ ;
$q_1 q_2 \dots q_n \leftarrow p_1 p_2 \dots p_n$ ;
**for** $k \leftarrow 2$ **to** $i/2$ **do**
  $a \leftarrow (j+k) \bmod n$ ;
  $b \leftarrow (j+i-k-1) \bmod n$ ;
  $c \leftarrow q_a$ ;
  $q_a \leftarrow q_b$ ;
  $q_b \leftarrow c$ ;
**end**
$p_1 p_2 \dots p_n \leftarrow q_1 q_2 \dots q_n$ ;

**Algorithm 1:** Pseudocode of $BC_{2,1}O$

```
input : N (the size of population),
        P (initial population),
        p^i (i-th solution in P),
        q (optimum solution)
output: P (new population)
p* ← Fittest(P) ;
while p* ≠ q and time < 100 do
    Q ← P ;
    for i ← 1 to N do
        a ← Mutate(p^i) ;
        Add(Q, a) ;
    end
    Sort(Q) ;
    R (new empty population) ;
    n ← 1 ;
    Add(R, q^n) ;
    i ← 2 ;
    while n < N do
        if q^i ≠ r^n then
            n ← n + 1 ;
            Add(R, q^n) ;
        end
        i ← i + 1 ;
    end
    for i ← 1 to N do
        if CanBeImproved(r^i) then
            r^i ← BC_{2,1}O(r^i) ;
        end
    end
    P ← R ;
    p* ← Fittest(P) ;
end
```

**Algorithm 2:** Pseudocode of The Genetic Algorithm with $BC_{2,1}O$

## IV. Results and Discussions

In this research, we use 50 problems from TSPLIB. The smallest one is burma14 that has 14 nodes and the biggest one is gr202 that has 202 nodes. We compare the performances of GA (genetic algorithm without $BC_{2,1}O$), $IABC_{2,1}O$ (iterative approach of $BC_{2,1}O$), and $GABC_{2,1}O$ (genetic algorithm with $BC_{2,1}O$). For $IABC_{2,1}O$, we use $12\ldots n$ as its initial solution. For GA and $GABC_{2,1}O$, the size of population used is 100.

For every problem, we test GA, $IABC_{2,1}O$, and $GABC_{2,1}O$ three times and pick the best solution (the solution with smallest distance). This test is done using the Java programming language on Netbeans IDE. The computer use an Intel I5 Processor and 4GB RAM.

We show the results of GA, $IABC_{2,1}O$, and $GABC_{2,1}O$ in TABLE VII. The first column is the name of the problem. The second column is the best known distance for the problem. It is available online on http://elib.zib.de/pub/mp-testdata/tsp/tsplib/stsp-sol.html. The third, fifth, and seventh column are the distance obtained by GA, $IABC_{2,1}O$, and $GABC_{2,1}O$ respectively. The fourth, sixth, and eighth column are the accuracy of the distance obtained by GA, $IABC_{2,1}O$, and $GABC_{2,1}O$ respectively. The accuracy is calculated by

$$\left(1-\frac{d_i-d_i^*}{d_i^*}\right)100\% \tag{5}$$

where $1 \leq i \leq 50$, $d_i$ is the distance of $i$-th problem obtained by GA, $IABC_{2,1}O$, or $GABC_{2,1}O$ and $d_i^*$ is the best known distance of $i$-th problem.

From those 50 problems, the average of the accuracy from GA, $IABC_{2,1}O$, and $GABC_{2,1}O$ are 65.12%, 94.21%, 99.82% respectively. We can see that for every problem, the distance obtained by $GABC_{2,1}O$ is less than or equal to the distances obtained by GA and $IABC_{2,1}O$. It can also be seen in the table, the distances obtained by $GABC_{2,1}O$ are equal to the best known distances for 37 different problems.

## V. Conclusions and Future Works

In this research, we proposed a genetic algorithm with $BC_{x,y}O$ for the traveling salesman problem. The idea of $BC_{x,y}O$ is to find the best combination of $x$ disjoint sub-solutions (also the reverse of sub-solutions) from $y$ known solutions.

In this research, we only use $BC_{2,1}O$. It is the simplest and the fastest one. It is still challenging to find $BC_{x,y}O$ simplification for $x \geq 3$ or $y \geq 2$. We are sure that better results will be obtained if we use bigger value of $x$ and $y$.

## References


[1] M. L. Shahab, D. B. Utomo, and M. I. Irawan, "Decomposing and solving capacitated vehicle routing problem (cvrp) using two-step genetic algorithm (tsga)," *Journal of Theoretical & Applied Information Technology*, vol. 87, no. 3, 2016.

[2] U. Hacizade and I. Kaya, "Ga based traveling salesman problem solution and its application to transport routes optimization," *IFAC-PapersOnLine*, vol. 51, no. 30, pp. 620–625, 2018.

[3] Muren, J. Wu, L. Zhou, Z. Du, and Y. Lv, "Mixed steepest descent algorithm for the traveling salesman problem and application in air logistics," *Transportation Research Part E: Logistics and Transportation Review*, vol. 126, pp. 87–102, 2019.

[4] A. S. Elgesem, E. S. Skogen, X. Wang, and K. Fagerholt, "A traveling salesman problem with pickups and deliveries and stochastic travel times: An application from chemical shipping," *European Journal of Operational Research*, vol. 269, no. 3, pp. 844–859, 2018.

[5] A. Fischer, F. Fischer, G. Jäger, J. Keilwagen, P. Molitor, and I. Grosse, "Exact algorithms and heuristics for the quadratic traveling salesman problem with an application in bioinformatics," *Discrete Applied Mathematics*, vol. 166, pp. 97–114, 2014.

[6] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook, *The traveling salesman problem: a computational study*. Princeton university press, 2006.

[7] M. Held and R. M. Karp, "A dynamic programming approach to sequencing problems," *Journal of the Society for Industrial and Applied mathematics*, vol. 10, no. 1, pp. 196–210, 1962.

[8] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Operations research*, vol. 21, no. 2, pp. 498–516, 1973.

[9] M. L. Shahab, "New heuristic algorithm for traveling salesman problem," in *Journal of Physics: Conference Series*, vol. 1218, no. 1, 2019, p. 012038.

[10] P. Larranaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, and S. Dizdarevic, "Genetic algorithms for the travelling salesman problem: A review of representations and operators," *Artificial Intelligence Review*, vol. 13, no. 2, pp. 129–170, 1999.

[11] G. Reinelt, "Tspliba traveling salesman problem library," *ORSA Journal on Computing*, vol. 3, no. 4, pp. 376–384, 1991.

TABLE VII: The results of GA, $IABC_{2,1}O$, and $GABC_{2,1}O$

| Problem | Best Known Distance | GA | | $IABC_{2,1}O$ | | $GABC_{2,1}O$ | |
|---|---|---|---|---|---|---|---|
| | | Distance | Accuracy | Distance | Accuracy | Distance | Accuracy |
| burma14 | 3323 | 3323 | 100 | 3461 | 95.85 | 3323 | 100 |
| ulysses16 | 6859 | 6859 | 100 | 7076 | 96.84 | 6859 | 100 |
| gr17 | 2085 | 2090 | 99.76 | 2211 | 93.96 | 2085 | 100 |
| gr21 | 2707 | 2707 | 100 | 2801 | 96.53 | 2707 | 100 |
| ulysses22 | 7013 | 7013 | 100 | 7163 | 97.86 | 7013 | 100 |
| gr24 | 1272 | 1272 | 100 | 1278 | 99.53 | 1272 | 100 |
| fri26 | 937 | 959 | 97.65 | 937 | 100 | 937 | 100 |
| bayg29 | 1610 | 1610 | 100 | 1686 | 95.28 | 1610 | 100 |
| bays29 | 2020 | 2048 | 98.61 | 2108 | 95.64 | 2020 | 100 |
| dantzig42 | 699 | 766 | 90.41 | 699 | 100 | 699 | 100 |
| swiss42 | 1273 | 1390 | 90.81 | 1410 | 89.24 | 1273 | 100 |
| att48 | 10628 | 10937 | 97.09 | 11045 | 96.08 | 10628 | 100 |
| gr48 | 5046 | 5627 | 88.49 | 5278 | 95.4 | 5046 | 100 |
| hk48 | 11461 | 12180 | 93.73 | 11718 | 97.76 | 11461 | 100 |
| eil51 | 426 | 463 | 91.31 | 460 | 92.02 | 426 | 100 |
| berlin52 | 7542 | 8297 | 89.99 | 8492 | 87.4 | 7542 | 100 |
| brazil58 | 25395 | 29586 | 83.5 | 27397 | 92.12 | 25395 | 100 |
| st70 | 675 | 765 | 86.67 | 712 | 94.52 | 675 | 100 |
| eil76 | 538 | 602 | 88.1 | 587 | 90.89 | 538 | 100 |
| pr76 | 108159 | 129164 | 80.58 | 121232 | 87.91 | 108159 | 100 |
| gr96 | 55209 | 71200 | 71.04 | 58601 | 93.86 | 55209 | 100 |
| rat99 | 1211 | 1503 | 75.89 | 1257 | 96.2 | 1211 | 100 |
| kroA100 | 21282 | 29754 | 60.19 | 22926 | 92.28 | 21282 | 100 |
| kroB100 | 22141 | 29938 | 64.78 | 24237 | 90.53 | 22141 | 100 |
| kroC100 | 20749 | 28149 | 64.34 | 22773 | 90.25 | 20749 | 100 |
| kroD100 | 21294 | 28303 | 67.08 | 23268 | 90.73 | 21294 | 100 |
| kroE100 | 22068 | 33621 | 47.65 | 23401 | 93.96 | 22068 | 100 |
| rd100 | 7910 | 9859 | 75.36 | 8607 | 91.19 | 7910 | 100 |
| eil101 | 629 | 762 | 78.86 | 699 | 88.87 | 629 | 100 |
| lin105 | 14379 | 21847 | 48.06 | 14962 | 95.95 | 14379 | 100 |
| pr107 | 44303 | 77314 | 25.49 | 47706 | 92.32 | 44303 | 100 |
| gr120 | 6942 | 9713 | 60.08 | 7475 | 92.32 | 6942 | 100 |
| pr124 | 59030 | 106506 | 19.57 | 63234 | 92.88 | 59030 | 100 |
| bier127 | 118282 | 156361 | 67.81 | 124191 | 95 | 119566 | 98.91 |
| ch130 | 6110 | 8514 | 60.65 | 6524 | 93.22 | 6139 | 99.53 |
| pr136 | 96772 | 151803 | 43.13 | 102668 | 93.91 | 97324 | 99.43 |
| gr137 | 69853 | 111523 | 40.35 | 71883 | 97.09 | 69853 | 100 |
| pr144 | 58537 | 115406 | 2.85 | 58812 | 99.53 | 58537 | 100 |
| ch150 | 6528 | 9892 | 48.47 | 7037 | 92.2 | 6554 | 99.6 |
| kroA150 | 26524 | 38934 | 53.21 | 28665 | 91.93 | 26620 | 99.64 |
| kroB150 | 26130 | 42767 | 36.33 | 28289 | 91.74 | 26141 | 99.96 |
| pr152 | 73682 | 185734 | -52.08 | 77039 | 95.44 | 73826 | 99.8 |
| u159 | 42080 | 66668 | 41.57 | 42981 | 97.86 | 42080 | 100 |
| si175 | 21407 | 24354 | 86.23 | 21570 | 99.24 | 21414 | 99.97 |
| brg180 | 1950 | 3840 | 3.08 | 1990 | 97.95 | 1950 | 100 |
| rat195 | 2323 | 3754 | 38.4 | 2397 | 96.81 | 2347 | 98.97 |
| d198 | 15780 | 26498 | 32.08 | 16692 | 94.22 | 15855 | 99.52 |
| kroA200 | 29368 | 51289 | 25.36 | 31231 | 93.66 | 29826 | 98.44 |
| kroB200 | 29437 | 47128 | 39.9 | 31853 | 91.79 | 29929 | 98.33 |
| gr202 | 40160 | 58869 | 53.41 | 43012 | 92.9 | 40555 | 99.02 |