The tables below show my results for the various algorithms and representations (where RR = repeated random, HC = hill climbing, P.P. = pre-partitioning, etc.). All values are the average over 100 randomly generated instances.

*iterations = 25,000*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | KK | RR stan. | RR p.p. | HC stan. | HC p.p. | SA stan. | SA p.p. |
| *residue (avg.)* | 254,357 | 297,810,699 | 135 | 372,424,438 | 1,134 | 342,312,788 | 186 |
| *time [s]*  *(avg.)* | 0.002 | 3.273 | 60.236 | 1.214 | 57.607 | 4.412 | 45.795 |

*iterations = 2,500*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | KK | RR stan. | RR p.p. | HC stan. | HC p.p. | SA stan. | SA p.p. |
| *residue (avg.)* | 209,967 | 23,621,629,689 | 17,796 | 9,163,891,569 | 26,016 | 8,994,478,430 | 20,490 |
| *time [s]*  *(avg.)* | 0.002 | 0.032 | 0.573 | 0.0116 | 0.543 | 0.041 | 0.577 |

From these results we can make the following observations.

1. Pre-partitioning led to solutions with dramatically lower residues, when compared to standard representation.
2. In no case did a randomized heuristic with standard representation return results lower than, or even comparable to, those returned by Karmarkar-Karp.
3. Using pre-partitioning, all three randomized heuristics gave substantially better solutions than did Karmarkar-Karp.
4. Pre-partitioning led to substantially longer runtimes for all three randomized heuristics.
5. In the pre-partitioned representation, increasing the number of iterations improved results from repeated-random and simulated annealing more than it did for hill-climbing.

The single best performance in both cases, in terms of residue, was repeated-random with pre-partitioning. In the case of 25,000 iterations, however, this method also took the longest. Meanwhile, simulated annealing with pre-partitioning gave results within 50% of repeated-random, while taking about 25% less runtime. At lower iterations, however, repeated-random did not take significantly longer than the other two randomized heuristics, and still gave the best results. Karmarkar-Karp, meanwhile, was in both cases the fastest method – by one to several orders of magnitude.

These results illustrate some interesting trade-offs among the various heuristics and representations. If one wants a solution very quickly and only requires that it be a *fairly* good one, Karmarkar-Karp may be the best choice. If better solutions are needed and there is more time to trade off, a randomized heuristic with pre-partitioning may be more useful. If one needs *very* good solutions and does not mind waiting a long time for them, a high iteration implementation of either repeated-random or simulated annealing (both with pre-partitioning) may be the best tool to use.