Comparative Analysis of the Sailor Assignment Problem

Joseph Vannucci, Deon Garrett, Dipankar Dasgupta
Department of Computer Science
University of Memphis
Memphis, TN 38152
{jvannucc,jdgarrtt,dasgupta}@memphis.edu

ABSTRACT

In this work the performance of several local search and metaheuristic methods is compared to previously reported work using evolutionary algorithms. The results show that while multiple algorithms are competitive on the Sailor Assignment Problem, the state-of-the-art evolutionary algorithm and a simulated annealing algorithm tend to provide the best performance. Additionally, some relevant features of the Sailor Assignment Problem are analyzed and used to explain the observed performance characteristics.

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: Constrained optimization

General Terms

Algorithms

Keywords

Genetic Algorithms, Assignment Problem, Metaheuristics

1. INTRODUCTION

Formally, the Sailor Assignment Problem [3] can be formulated as an assignment problem matching a each of N sailors to one of a group of M jobs, while maximizing a utility function [2]:

$$\sum_{i=1}^{N}\sum_{j=1}^{|M|}\mathcal{F}_{i,j}d_{i,j},$$

subject to:

$$\sum_{i=1}^{N} d_{i,j} \leq 1, \quad \forall j \in \{1,2,\ldots,M\}$$

and

$$\sum_{j=1}^{M} d_{i,j} \leq 1, \quad \forall i \in \{1,2,\ldots,N\},$$

where

$$d_{i,j} = \begin{cases} 1 & : \text{ Sailor } i \text{ assigned to job } j \\ 0 & : \text{ otherwise} \end{cases}$$

GECCO'06, July 8–12, 2006, Seattle, Washington, USA.
ACM 1-59593-186-4/06/0007.

$\mathcal{F}$ is the fitness matrix defining how well suited a match is, and d is a binary value that denotes if a sailor is actually assigned to that job. This definition addresses logistical constraints, one or fewer sailors can be assigned to each job and one or fewer jobs can be assigned to each sailor.

2. APPROACHES TO THE SAP

The methods used are a random-restart, steepest-ascent hill-climber, which is used for comparison and analytical purposes. A simulated annealing algorithm was tested with various annealing schedules. Our experiments show that the best parameters for simulated annealing t_0 is .002, α is .99,and i is 2000. A tabu search with varying tabu tenures was also tested. Experiments show that using a tenure between $.1 - .2 \times problemsize$ is best. CHC [1] was tested against and we used the parameters specified in [2].

3. SAP NEIGHBORHOOD FUNCTIONS

In previous work with Genetic Algorithms [2] a repair mechanism is required to ensure that crossover produces feasible individuals. The repair mechanism is somewhat expensive, possibly providing an advantage in the use of local search procedures which can avoid this type of constraint violation.

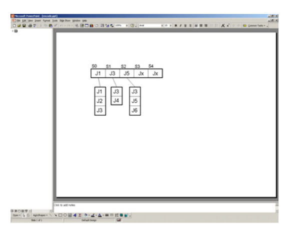

Figure 1: Visualization of the SAP as a chromosome used by genetic algorithms and iterative search. An example one-swap mutation is to give Sailor 0 Job 2. An example of two-swap is to give Sailor 0 Job 3 and Sailor 1 Job 4.

The neighborhood functions available in this implementation are informally called 1X and 2X. 1X is nothing more than looking at a particular sailor and trying to assign him another job which has no conflicts. 2X is basically the same idea but more powerful. It first chooses a sailor S_1 and tries to reassign him to a new, non-conflicting job J_1. If that is not possible it will choose a conflicting job J_2 and attempt to reassign the sailor S_2 currently assigned to J_2, by finding

Table 1: Baseline comparisons for the four algorithms on two problem sizes.

Problem Size	CHC mean	CHC std. dev.	SA mean	SA std. dev.	TS mean	TS std. dev.	RR mean	RR std. dev.
500	0.6806	0.0004	0.6805	0.0003	0.6738	0.0023	0.6672	0.0037
1000	0.6769	0.0007	0.6790	0.0018	0.6674	0.0018	0.6652	0.0002

a job J_3 that is available to S_2 and not taken by any other sailor. If this is successful, it will reassign S_2 to J_3 and S_1 to J_2. If no such job J_3 can be found, the proposed move fails and another sailor is selected for reassignment.

The expected neighborhood size is as follows. Then

$$\mathbf{E}[X] = \left(1 - \frac{n}{m}\right) |\mathcal{N}_{1X}(x)| + \left(\frac{n}{m}\right) |\mathcal{N}_{2X}(x)|,$$

where $|\mathcal{N}_{1X}(x)|$ is the number of solutions that must be examined to take a 1X move from solution x. Because the $1-\frac{n}{m}$ term ensures that the 1X move is feasible, $|\mathcal{N}_{1X}(x)| = 1$, since we need only to accept the feasible move.

On the other hand, after we learn we must apply the 2X operator, there are $k-1$ possible ways to perform this reassignment. Thus, $|\mathcal{N}_{2X}(x)| = k-1$. Additionally, these calculations assume that we have selected a single job to assign to the selected sailor. In reality, there are $k-1$ possible jobs we could choose from for each sailor, and n possible sailors to choose from. Thus, the actual expected number of neighbors we may need to examine is given by

$$\mathbf{E}[|\mathcal{N}(x)|] = n\sum_{i=1}^{k-1}\left(1 - \frac{n}{m}\right) 1 + \left(\frac{n}{m}\right)(k-1)$$

$$= n(k-1)\left(1 - \frac{n}{m}\right) + n\,(k-1)^2\left(\frac{n}{m}\right).$$

For example, when we have $n = 1000$, $m = 1100$, and $k = 5$ we may expect to have to examine approximately 15,000 points in order to determine which is the best move. Because the tabu search algorithm must examine all these neighbors at each iteration, it requires an enormous number of evaluations to reach high quality solutions.

4. ANALYZING THE SAP

An important indicator of the performance of metaheuristics is the distribution of local optima. One method to attempt to obtain information of this sort is by calculating the entropy of the local optima. Taillard [4] finds that in completely random instances of the QAP running a robust tabu search some large number of times produces solutions with an an entropy of roughly .97, but when the same number of tabu searches are run on a real-world QAP instance the entropy was considerably lower, 0.8. The entropy calculated over a random SAP instance is .79, which shows that even random instances of the Sailor Assignment Problem are more structured than random instances of the QAP.

The entropy of the set with respect to a given sailor is a measurement of uniformity over his possible jobs, while the entropy of the entire solutions is the averaged entropy of all the sailors.

$$E_i = \frac{-\sum_{j=1}^{c}(\frac{n_{ij}}{m})log(\frac{n_{ij}}{m})}{log(c)}.$$

The equations sums over all jobs j in M. The important difference between Aarts' formulation and our is the value c. In the QAP n is used because each position is eligible for each possible entry, it is a permutation. Whereas in the SAP each position is not eligible for each entry. Thus we must divide by $log_2(c)$ instead of $log_2(n)$. To calculate the entropy over the entire set of solutions we average the entropy of each sailor:

$$E = \frac{\sum_{i=1}^{n} E_i}{n}$$

where n is the number of sailors.

5. RESULTS AND CONCLUSIONS

The results in Table 1 display the comparative results of 30 trials each. CHC and Simulated Annealing perform equally well, while tabu search and random restart hillclimbing are inferior. The obvious difference is that tabu search and hillclimbing is steepest-accent (or best-accent) while CHC and simulated annealing are both probabilistic in nature. In steepest accent, evaluations would very often be wasted early. Also, weak basins of attraction would aggravate this. If there are many local optima tabu search would spend significant computational time to climb directly to the top of a basin and then to leave this basin, which would quickly exhaust its evaluations. CHC is perhaps aided by the fact that is will quickly converge to a good region of the search space and explore it quite thoroughly. After it hones in a small region of the space it will them randomly restart and quickly converge to a very different region of the space.

It is the hope of the authors that the tools derived for the SAP will aid any in future research. The addition of an adaptive annealing schedule which integrates domain knowledge would likely aid the performance of the simulated annealing algorithm, perhaps besting CHC given their similar performance. Tabu search does not seem to be a good choice because of the very large neighborhood space.

6. REFERENCES

[1] L. Eshelman. The CHC Adaptive Search Algorithm. In *Foundations of Genetic Algorithms I*, pages 265–283. Morgan Kaufmann, 1991.

[2] D. Garrett, J. Vannucci, R. Silva, D. Dasgupta, and J. Simien. Genetic Algorithms for the Sailor Assignment Problem. In *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation*. ACM Press, New York, NY, 1921-1928. 2005

[3] J. Liebowitz and J. Simien. Computational Efficiencies for Multi-Agents: A Look at the NPRST's Multi-Agent System for Sailor Assignment. 2004.

[4] E. D. Taillard, Comparison of Iterative Searches for the Quadratic Assignment Problem. Location science 3, 1995, 87-105.