Plateau Connection Structure and Multiobjective Metaheuristic Performance

Deon Garrett
Department of Computer Science
University of Memphis
Memphis, TN, 38152
Email: deong@acm.org

***Abstract*—This paper proposes the plateau structure imposed by the Pareto dominance relation as a useful determinant of multiobjective metaheuristic performance. In essence, the dominance relation partitions the search space into a set of equivalence classes, and the probabilities, given a specified neighborhood structure, of moving from one class to another are estimated empirically and used to help assess the likely performance of different flavors of multiobjective search algorithms. The utility of this approach is demonstrated on a number of benchmark multiobjective combinatorial optimization problems. In addition, a number of techniques are proposed to allow this method to be used with larger, real-world problems.**

I. Introduction

The use of metaheuristics for multiobjective optimization problems has gained tremendous popularity over the past several years. In particular, multiobjective evolutionary algorithms have become exceptionally popular in this area, largely due to the natural fit of a population-based search to problems involving finding a representative set of Pareto optimal solutions. However, in many combinatorial optimization problems, the performance of evolutionary algorithms can be improved through the inclusion local search methods, and in some cases, the local search based metaheuristics outperform the evolutionary techniques in their own right.

It is thus important that we begin to develop insights into how each new metaheuristic searches the space, and how these traits map onto performance across different types of search spaces and their implicit structures. While there has been some work in the past on this question, we are still far from definitive answers. Toward that end, this paper will discuss the measurement and estimation of a particular type of structure imposed by a given multiobjective optimization problem, the Pareto plateau structure, and examine its impact on different classes of multiobjective metaheuristics. Section II will describe the concept of Pareto plateaus and discuss how the plateau structure of a given problem is obtained. Interpretation of the resulting data is discussed in Section III. Section IV describes the empirical support for the hypothesis put forth in this work, and the conclusions are presented Section V along with several directions for continued research.

II. Pareto Plateaus and Problem Difficulty

One of the most important tasks remaining for researchers working with metaheuristics for multiobjective optimization is to begin to formalize the notions that not all algorithms perform equally on a given problem and to better understand the source of these differences in efficacy. As in any search algorithm, the performance of these methods depends on the interaction between the guiding features of the search algorithm and the particular structures inherent in a given optimization problem.

There have been a few attempts to characterize landscape features in multiobjective problems. Knowles and Corne [11] proposed a multiobjective variant of the quadratic assignment problem and examined some static features of the landscapes. In addition, they conjectured that certain features of the landscapes might cause the problem to lend itself to certain types of optimization algorithms.

Subsequently, the present author considered the dynamics of typical search algorithms and their effect on performance on problems with particular landscape features [8], [9], [7]. This line of research was focused on modeling not only the properties of the landscapes themselves, but also on the interaction between search algorithms and the various landscape characteristics.

In this work, we examine another such property – the plateau structure imposed by the Pareto dominance relation on the particular problem. Plateau Connection Graphs (PCGs) are not unknown in the literature. Hoos [10] described their use in modeling MAX-SAT problems, for example. The basic idea is to simply aggregate all solutions with equal fitness into a single state, and then consider the probabilities of moving from one state to another. The result can be viewed as a weighted directed graph where each node represents a single equivalence class and the weights are given by the probabilities of moving between two classes.

In the case of single objective optimization problems, the presence or absence of plateau structure is intimately bound to the properties of the particular problem. For example, MAX-SAT problems exhibit very large scale plateau structure due to the limited number of unique fitness values available and the inherent effects of flipping the value of a term. However, when the notion is generalized to the multiobjective realm, we see that essentially *every* problem exhibits this sort of plateau structure, as fitness is no longer assigned purely as a function of the objective functions, but instead as a function of the dominance relation between individual solutions.

Indeed, multiobjective evolutionary algorithms such as NSGA-II [5] explicitly incorporate this property into the core of the algorithm.

Thus, more so than in conventional optimization, plateau structure would seem to play a major role in a wide variety of multiobjective optimization problems. The performance of many metaheuristics depends crucially on the ability of the search to navigate the search space efficiently. The first requirement is that the algorithms be able to locate points on or very near the Pareto front itself. Because most such methods rely on random or heuristically generated starting locations likely to be far from the front, the degree to which the algorithm can move from high rank nodes to low ranked nodes is very important. Stated in terms of the PCG, algorithms such as these rely for their efficacy on the presence of available exits from each plateau to successively better ones.

In addition, the algorithms must arrange to find a reasonably uniform covering of the potentially wide range of Pareto optimal solutions. Here, there exist multiple strategies. First, an algorithm may simply run repeated searches from different starting locations in an attempt to build up the diverse Pareto set approximation one solution at a time. Another option, typically taken by evolutionary algorithms, is to attempt to cover the front in one single run. Yet a third option is to attempt to build a diverse Pareto set approximation by finding a single Pareto optimal point, then trying to gradually sweep that point along the Pareto front until a suitably diverse set of solutions has been found.

Each of these techniques imposes a different trajectory through the search space. Through careful examination of the properties of the space, it may be possible to determine *a priori* which type of approach would be the most fruitful. The plateau structure imposed by the dominance relation can tell us much about the relative difficulty of a local search algorithm to reach the desired set of solutions through each type of trajectory. This paper is focused on the question of how best to obtain the information about the plateau structure and the proper framework for analysis of the resulting information.

III. Interpretation of Plateau Structure

Perhaps the simplest method of obtaining useful information from the plateau connection structure is simply to visually inspect the resulting graphs. Each "level" of nondominance is represented by a node in the graph, and the edges between nodes in the directed graph are weighted to indicate the probability of moving from the source node to the destination node via purely random moves under the proposed neighborhood. Fig.1 shows a simple example of such a graph.

Note that in Fig.1, as well as throughout the remainder of this paper, the following nomenclature shall be adopted. Each node in the graph will be labeled as Rx, where x is a nonnegative integer denoting the rank of the equivalence class represented by that node. By convention, $R0$ will denote the true Pareto front, with $R1...Rn$ representing dominance ranks moving steadily away from the front. Where applicable, the last rank Rn will denote infeasible solutions to constrained problems.

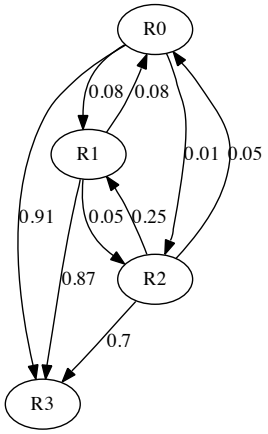

Fig. 1. Sample of a Pareto Plateau Connection Graph. The node label Rx indicates nondomination level x, where $x = 0$ indicates the true Pareto Front.

A. Coarse Graining

One very real problem in analyzing the resulting data is the very large scale of the resulting graphs and Markov models. In these cases, it may prove useful to provide more focused windows onto the data. One method by which this may be performed is by coarse graining the nodes of the graph. By coarse graining, here it is meant that some number of adjoining fronts are coalesced into a state in the Markov chain. For instance, we may consider R0 to be not only the set of nondominated solutions, but also the set of solutions dominated by only the Pareto optimal solutions. By continuing this process for every unique front, the number of nodes in the graph will be halved with a corresponding nonlinear decrease in the number of edges.

This smaller graph provides strictly less information than the original. It is no longer possible to determine exactly the probability of moving from any front to an adjacent front. However, the reduction in the overwhelming detail of the original can help in obtaining the "big picture" overview.

B. Targeted Subgraphs

Another possibility for handling the explosion of states is to focus on a single specific region in the search space. For example, by considering only R0 and the nodes directly connected, we may obtain a complete picture of the neighborhood structure at the Pareto front. For even quite large problems, the number of states in this reduced form of the model is likely to remain small enough for reasonably simple analysis. By chaining together a number of this restricted models, we may begin to understand the structure of the search space as the search moves from initial regions of low fitness toward the true Pareto front.

IV. Experiments and Results

To demonstrate the utility of the proposed method of examining the plateau structure, a set of benchmark multiob-

jective combinatorial optimization problems were considered, including the Quadratic Assignment Problem, the Generalized Assignment Problem, and a multiobjective 0-1 Knapsack problem. For each problem, two approaches were considered. First, very small instances of the problems were exhaustively enumerated yielding an exact PCG. In addition, larger instances were considered in which the search space was estimated by taking a sample of points encountered by a stochastic local search algorithm. Together, these two approaches help to build a more complete picture of the structure of the search space, without the requirement of complete knowledge of the search space for realistic problems.

A. Multiobjective 0-1 Knapsack Problem

For one benchmark problem, we considered a multiobjective variant of the 0-1 knapsack problem. Like a conventional knapsack problem, we are given a list of items, each with an associated weight and value. The goal is to maximize the total value obtained without exceeding the weight capacity of the knapsack. To transform the problem into the multiobjective realm, each item was assigned multiple distinct values.

Formally, the problem is defined as follows. Given a set of n items with w_i denoting the weight of item i and v_i^m the value of item i with respect to the m^{th} objective function, find a binary string b of length n that minimizes the vector $\mathcal{F}$ given by

$$\mathcal{F}^m(b) = \sum_{i=1}^{n} b_i v_i^m \quad \forall m : 1 \leq m \leq k, \tag{1}$$

subject to

$$\sum_{i=1}^{n} b_i w_i \leq C, \tag{2}$$

where C is the capacity of the knapsack.

In the 0-1 versions of the knapsack problem, only one of each item is assumed to exist. Thus, a candidate solution takes the form of a simple binary string in which $b_i = 0$ implies that item i was not selected, and a value of 1 means that item i was selected. This naturally suggests the use of a binary string for encoding of candidate solutions within the search algorithm, as well as the use of standard search operators such as Hamming neighborhoods in local searchers, and multipoint or uniform crossover and bitwise mutation in evolutionary algorithms. This is unlike the mQAP and mGAP problems which require permutation-based and constrained integer encodings to most naturally represent solutions to the problems. Thus, this choice of three benchmark problems provides a wide range of encoding types, as well as a fairly diverse set of landscape features.

Fig.2 shows the plateau connection graph for a small instance of the knapsack problem. This instance contains 20 items, five objectives, and features very tight constraints so that most possible solutions are infeasible. We see from Fig.2 that there are nine distinct levels of dominance in the pool of candidate solutions, assuming that all infeasible solutions are considered to be dominated by all other solutions. Due to the tightness of the constraints, the weights on the edges terminating in $R8$, the infeasible region, are quite large. We can also begin to get an idea of the relative difficulty of approaching the Pareto front from an arbitrary starting location. For example, assuming we choose a random starting point that falls into the equivalence class $R3$, we see that with a probability of 9.7%, we will take a move into $R1$. From there, we have a 3.7% chance of reaching $R0$, the Pareto front, in one additional move.

Of course, most practical algorithms do not take a random move. However, these probabilities have direct bearing on the performance of algorithms such as simulated annealing, tabu search, next descent local search, and other techniques which do not necessarily exhaustively search the neighborhood before choosing their next move.

Another feature of Fig.2 illustrates a potential problem for certain types of multiobjective optimizers. So called *two-phase* local search algorithms which operate by attempting to move parallel to the front rely on ready pathways from one Pareto optimal solution to another. A cursory glance at the PCG reveals that there are no self loops at any node. Recall that the constraints for this problem were set quite tightly. With a Hamming neighborhood, any attempt to flip one bit of a solution already near the Pareto front is almost certain to either cause the solution to violate the weight constraint or reduce the value contained in the knapsack unnecessarily. What is needed in that case is not a single bit flip, but a swap of one selected item for another.

Fig.3 shows the PCG of the same problem using a swap neighborhood. Note that almost every node exhibits a self loop with a reasonably large weight. Also note that the probabilities of reaching $R8$ from near the Pareto front are significantly smaller. Again, given our knowledge of the knapsack problem, these results are in perfect alignment with our expectations.

Finally, the absence of $R7$ from the graph at all is an artifact of the way the graphs were generated. Only nodes which were connected to at least one other node were shown, and only moves that result in an actual change to the candidate solution were considered. As $R7$ consists of the single solution in which no items were selected, no swap operation can change the solution, and thus the node is unconnected to the rest of the graph.

However, also expected is the decrease in the probability of moving to a "good" front from a higher ranked front. Consider a candidate solution that has only one selected item in the knapsack. This solution may be far away from the Pareto front, yet the only option available is to swap the selected item for another. What is needed in this case is the ability to flip a single bit from a zero to a one when one is far from the front, and have the ability to swap two items when one is near the front. This is precisely the reason why algorithms such as Variable Depth Search (VDS) have been quite successful on knapsack problems. These algorithms allow the search to augment its strategy as the circumstances of the search require.

Fig.4 shows a final PCG for the example knapsack problem, this time utilizing a neighborhood which includes all possible

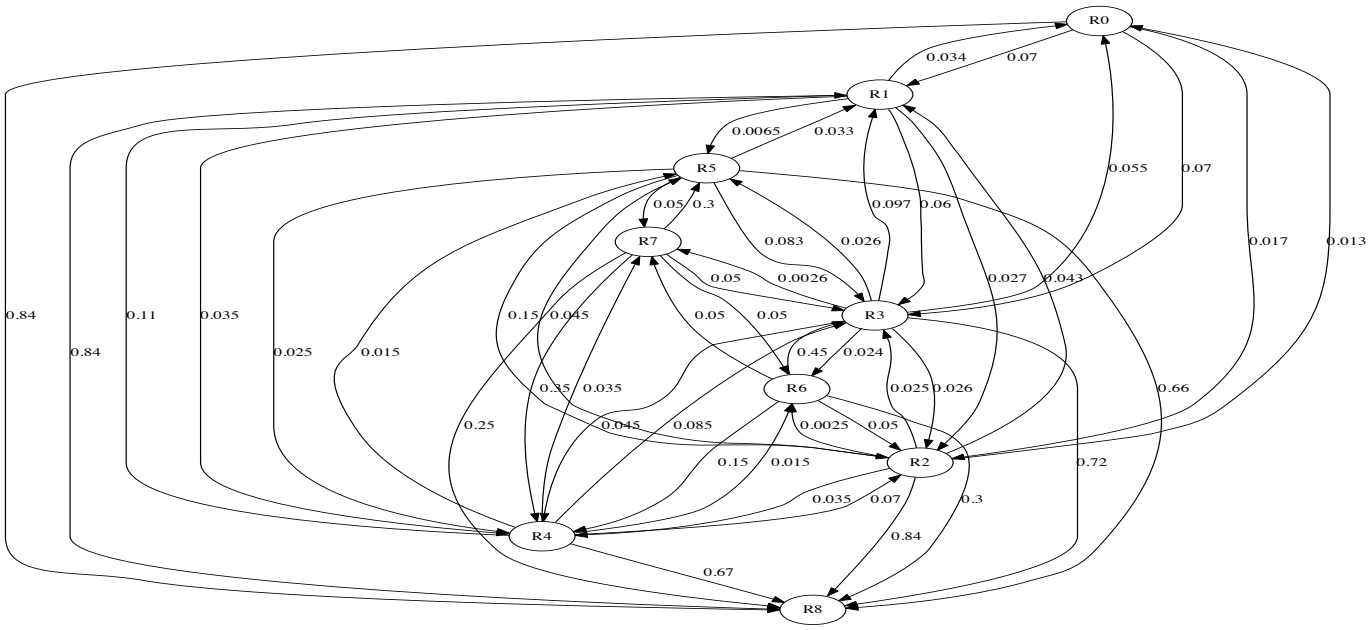

Fig. 2. Plateau connection graph for a 0-1 Knapsack problem with 20 items, five objectives, and very tight constraints using a simple Hamming neighborhood.

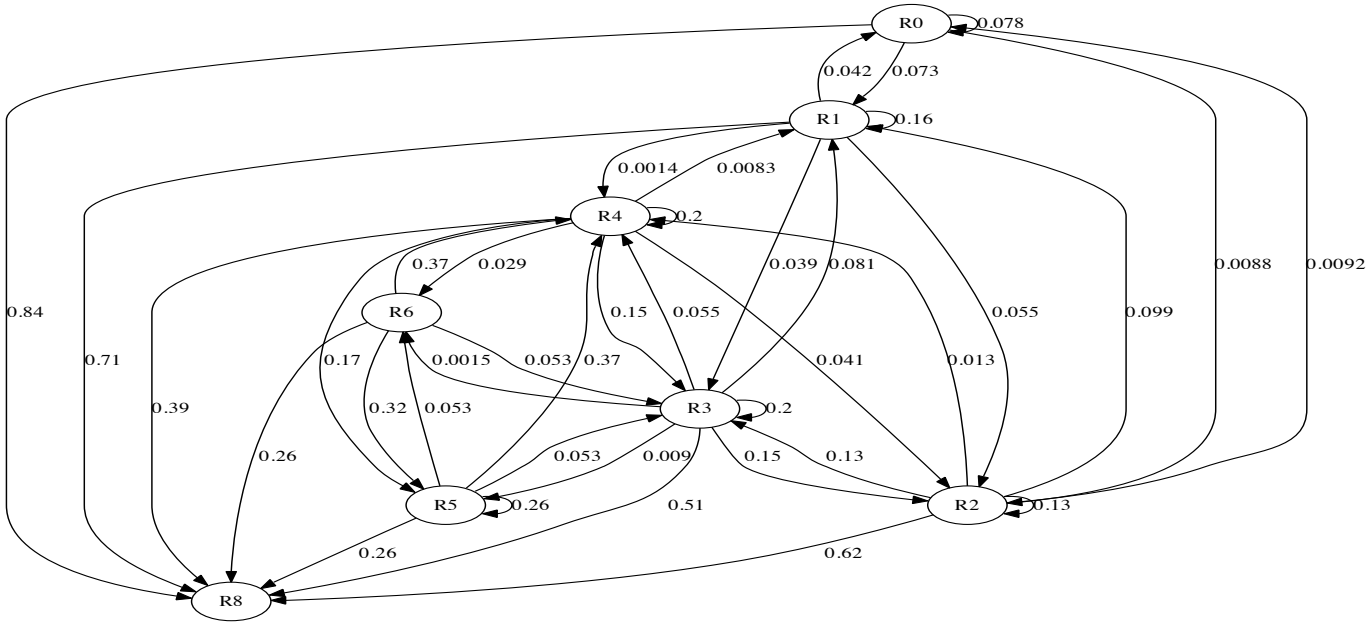

Fig. 3. Plateau connection graph for a 0-1 Knapsack problem with 20 items, five objectives, and very tight constraints using a swap neighborhood.

Hamming neighbors as well as all possible swap neighbors. As we can see, the graph does exhibit much of the structure that we expect from our a priori knowledge. It includes the self loops allowing the search to move along a single front, yet also shows significantly higher weights on the edges leading from poor solutions into ranks nearer the Pareto front. Note that this is a much larger neighborhood, so even if there are more exits from one rank to another, the weight along that edge may be smaller due to the larger number of total neighbors of that rank.

B. Multiobjective Quadratic Assignment Problem

The quadratic assignment problem (QAP) is one of the oldest and most widely studied combinatorial optimization problems. First formulated by Koopmans and Beckmann in 1957 [13], the QAP can be described as follows: Given two $n \times n$ matrices $\mathbf{A}$ and $\mathbf{B}$, find a permutation π that minimizes

$$\min_{\pi} \mathcal{F}(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{i,j} B_{\pi_i,\pi_j}. \tag{3}$$

Conventionally, the matrices $\mathbf{A}$ and $\mathbf{B}$ are called the *dis-*

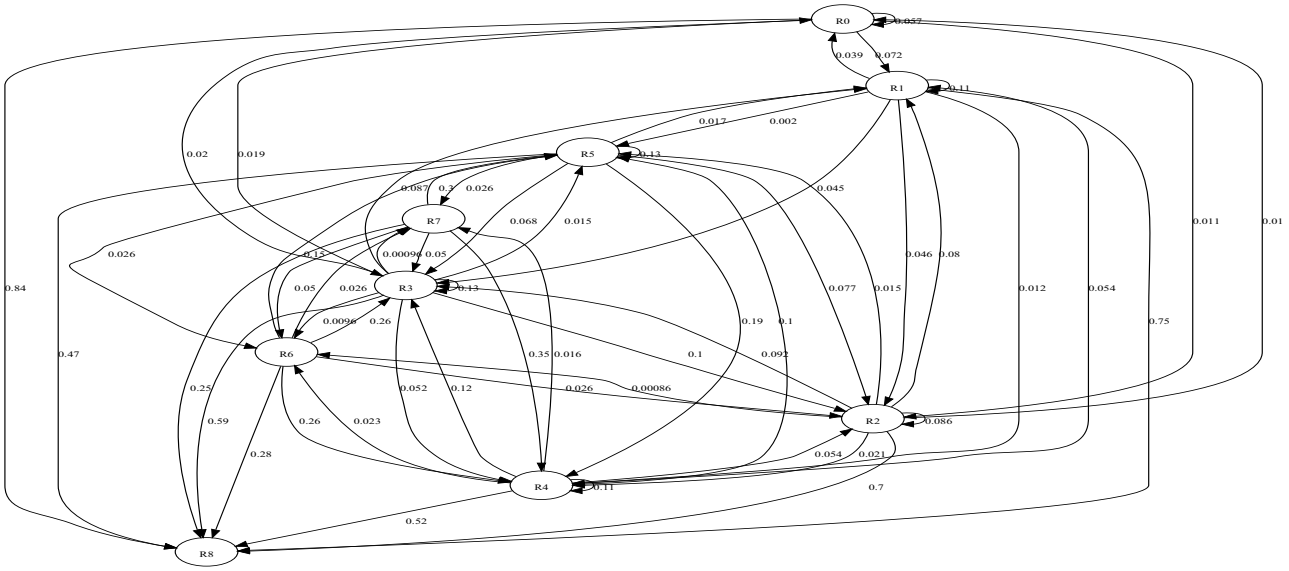

Fig. 4. Plateau connection graph for a 0-1 Knapsack problem with 20 items, five objectives, and very tight constraints using a neighborhood which allows either single bit flips or two bit swaps.

tance and *flow* matrices, the terminology arising from the original formulation of QAP as a facilities layout problem. Despite the terminology, QAP is useful in model several disparate application areas, including backboard wiring, hospital layout, and keyboard design. Not only $\mathcal{NP}$-hard [18], QAP is generally considered to be among the hardest optimization problems, with even relatively small instances posing a significant challenge to state-of-the-art branch and bound solvers. As shown by Sahni [18], there also exists no polynomial algorithm with a guaranteed error lower than some constant for every instance of QAP unless $\mathcal{P}$=$\mathcal{NP}$. As a result, stochastic local search (SLS) algorithms are the methods of choice for solving most large scale QAP instances.

In a pair of papers, Knowles and Corne proposed and provided detailed static analysis of the multiobjective QAP (mQAP) [11], [12]. The mQAP consists of a single $n \times n$ distance matrix, and k distinct $n \times n$ flow matrices. There exist then k different pairings of the distance matrix with one flow matrix, yielding k independent single objective QAP problems. The objective function value of a permutation π is thus a k-dimensional vector with

$$\mathcal{F}^m(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{i,j} B^m_{\pi_i,\pi_j} \quad \forall m : 1 \leq m \leq k. \qquad (4)$$

The mQAP models any sort of facilities layout problem in which the minimization of multiple simultaneous flows is required.

The mQAP has been studied by a number of researchers since its introduction. However, little has been done to examine the properties of the landscape and their effect on algorithm performance. Knowles and Corne provide a wealth of knowledge concerning a set of benchmark instances. However, their approach considers only static properties of the landscape, omitting the effects caused by the detailed behavior of the search algorithm.

In a rather comprehensive experimental study, Lopez et. al. [15] concluded that a memetic algorithm combining SPEA2 [24] and Robust Tabu Search [19] outperformed a wide variety of other metaheuristic approaches on a set of two-objective mQAP instances. In this paper, we examine the plateau structure of mQAP instances to better understand and support the experimental results reported in the literature.

Fig.5 shows the PCG for a very small instance of the mQAP. It shows that from the Pareto front, we have a 45% chance of remaining either in the front or in the next nearest rank. Another important feature is that very few of the weights in the PCG for the mQAP are as small as the smallest weights in either the knapsack or mGAP instances. In other words, there is a greater likelihood of making large jumps in the mQAP than in the other problems. This is in agreement with experimental results in [8] and elsewhere that state that the mQAP exhibits much less structure than many other combinatorial optimization problems.

Even with the small problem size, at only 12 nodes and 165 edges, we are rapidly approaching the point at which the graph no longer provides any usable information. In the mGAP example, we illustrated a simple way to reduce the complexity of the information provided by choosing to focus on a single restricted region of the search space. Here we will illustrate another technique which can help distill the complex information provided by the plateau structure into something more easily digested.

To illustrate, we generated another mQAP instance, identical to the first but with only two objectives instead of four.

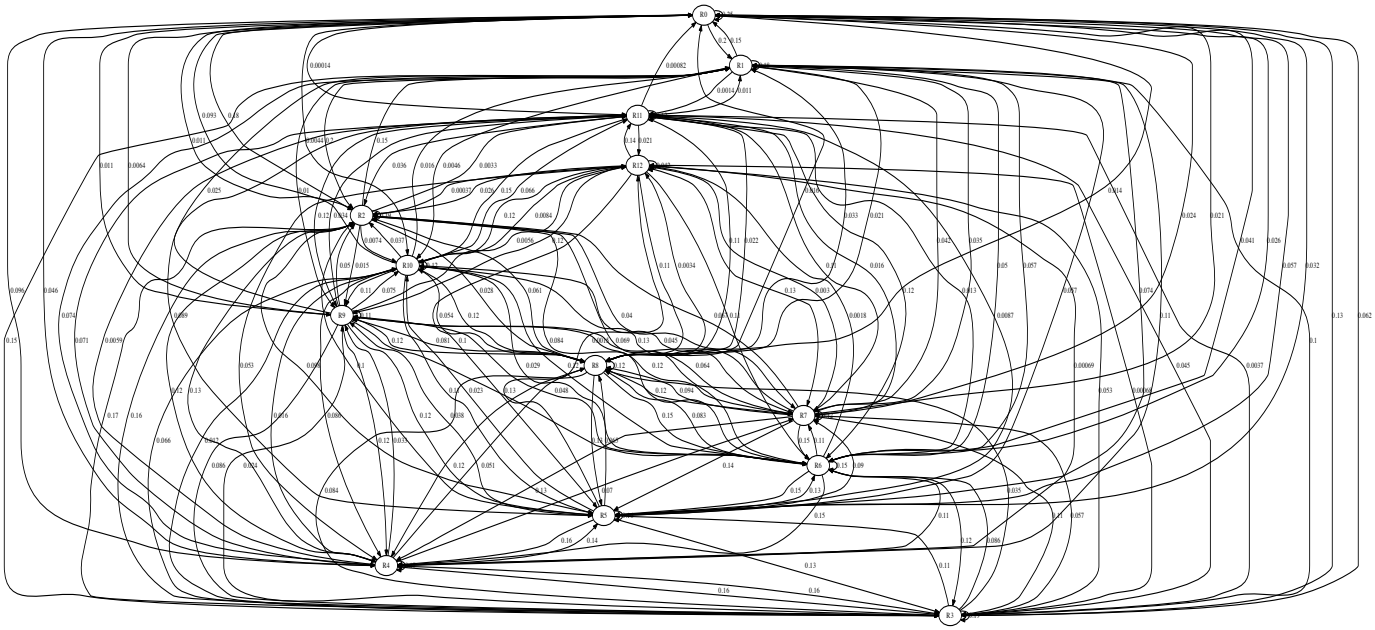

Fig. 5. PCG for a seven facility, four objective QAP instance.

As noted earlier, the decrease in the dimensionality of the objective space causes an explosion in the size of the PCGs. In this case, we obtain a PCG with 117 nodes and 11,469 edges. Coarse graining this graph by a factor of ten yields a new PCG, shown in Fig.6, with 11 nodes and 144 edges. While still relatively large, this graph is not so overwhelmingly large that no useful information can be obtained.

Examination of Fig.6 reveals that it is very similar to Fig.5. By combining this type of coarse grained picture with the targeted subgraphs described in the next section, it is possible to begin to gain a better understanding of even large and complex problems.

C. Multiobjective Generalized Assignment Problem

In contrast to the QAP, many classical combinatorial optimization problems possess what is sometimes called the Big Valley structure in which good local optima tend to be found near other good local optima [3]. This is in one sense the polar opposite of the QAP, and thus the QAP may not be an accurate representative of this sort of problem. Thus, we also chose to examine a problem exhibiting this type of structure, the multiobjective generalized assignment problem (mGAP), and to examine the impact of its more typical structure on the performance of multiobjective metaheuristic search algorithms.

The generalized assignment problem (GAP) deals with a set of m agents and a set of n tasks. Each task must be completed by exactly one agent. Each agent is allocated a specific number of resource units, and each agent requires a particular number of units to complete each task. Additionally, each agent incurs a specified cost for each task. The resource requirements and costs for a given task may differ between agents. The overall goal is to assign all tasks such that no agent violates the capacity constraints and the total costs incurred are minimized.

Formally, we may introduce an m-dimensional vector $\mathbf{B}$, with b_j denoting the total capacity alloted to agent j. We further introduce $m \times n$ matrices $\mathbf{A}$ and $\mathbf{C}$, denoting the resource matrix and the cost matrix respectively. Finally, we introduce an $m \times n$ binary matrix $\mathbf{X}$, with $x_{ij} = 1$ only if task i assigned to agent j by a particular candidate solution. The goal is thus to find such a solution so that

$$\min_{\mathbf{X}} \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} c_{ij}, \tag{5}$$

subject to

$$\sum_{i=1}^{m} x_{ij} a_{ij} \leq b_j \quad \forall\, j : 1 \leq j \leq n \tag{6}$$

and

$$\sum_{j=1}^{n} x_{ij} = 1 \quad \forall\, i : 1 \leq i \leq m, \tag{7}$$

where Equation (6) are known as the *capacity constraints*, and Equation (7) are called the *semi-assignment constraints*.

The GAP is known to be $\mathcal{NP}$-hard [18], and exact algorithms have proven tractable only for problems in the range of hundreds of tasks or less [22]. Thus, for large instances, heuristic and metaheuristic methods have received a great deal of attention, including tabu search approaches [6], [14], variable depth search [23], [17], ant colony optimization [16], evolutionary algorithms [4], and more recently, path relinking algorithms [1], [2], [20], [21].

Also in contrast to the mQAP, the mGAP is a constrained optimization problem. Constraints may be treated in a number

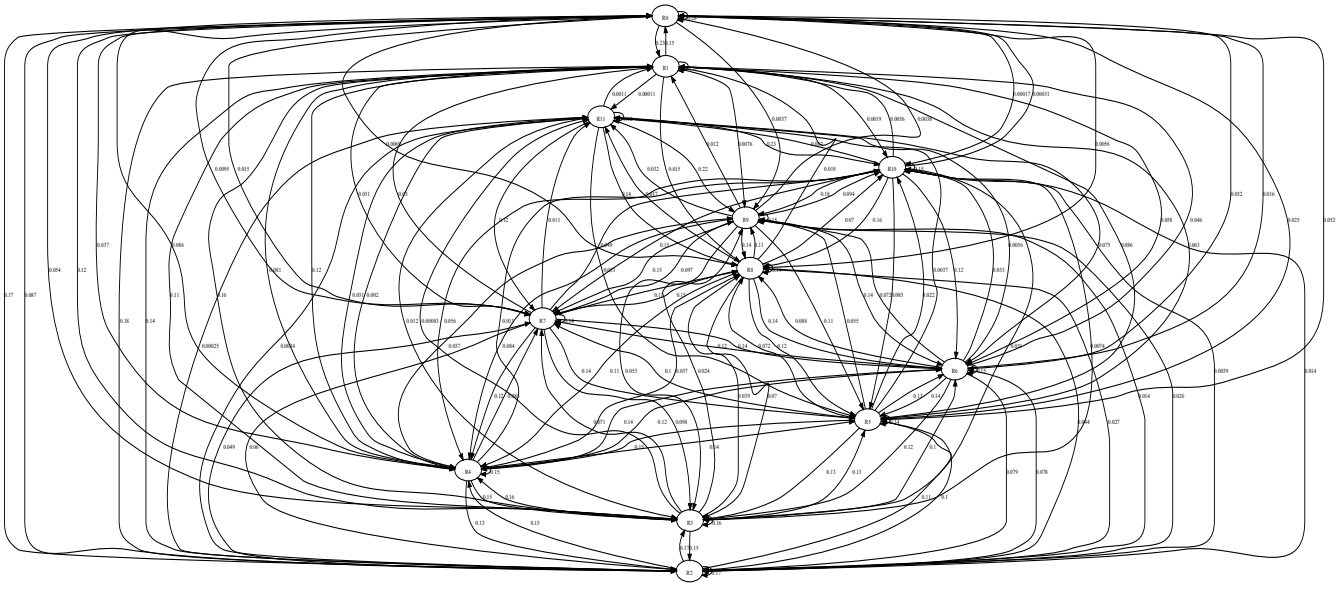

Fig. 6. PCG for a seven facility, two objective QAP instance coarse grained by a factor of ten. Thus, $R0$ in the graph represents all points in the ten equivalence classes nearest the Pareto front.

of ways when generating the PCGs. In this work, we treat infeasible solutions as completely tabu, and generate the graphs as though infeasible regions of the space are sinks from which a search algorithm cannot escape. This is not a terribly realistic assumption, but it suffices to point out that one could easily apply a penalty function, repair operator, or other custom behavior to infeasible solutions before determining their relative fitness, and thus constraints are quite independent of the analysis provided here.

Although we need not treat constraints specially, we can still learn useful information about the problem by handling violations in various ways. If, as in this work, all infeasible solutions are assigned to a single equivalence class, we may obtain an estimate of the tightness of the constraints. This information can help inform the design of algorithms which attempt to skirt the boundaries of the feasible regions where high quality solutions often lie.

To examine the structure of the mGAP, a number of small instances were generated. As one example, a 3×17 instance (three agents and 17 tasks) with two objectives was generated and exhaustively enumerated. For the generation of the PCG, a neighborhood function was used which allowed the search to try all possible *shifts*, i.e., a move of a single task from one agent to another, as well as all swaps involving the agents assigned to two tasks. As the number of objectives decreases, the number of distinct dominance ranks tends to grow much larger. Compared to the knapsack problem described previously which, with five objectives, exhibited fewer than ten nodes, the mGAP instance resulted in a PCG with 244 nodes and over 32,000 edges.

Clearly, we cannot draw a graph with this many edges in any meaningful way. One option, as described earlier, is to visualize meaningful subsets of the graph. Fig.7 shows the subset of the graph containing only $R0$ and all adjacent nodes. This indicates the structure of the search space at and around the Pareto front.

Much like the knapsack problem considered earlier, the presence of tight constraints strongly impacts the structure of the search space around the Pareto front. As shown in Fig.7, 95% of the neighbors reachable in one step from a point on the Pareto front are infeasible. Interestingly, none of the three nearest ranks to the true Pareto front appear at all in the graph, meaning that there is no move from any of the three next best domination levels to the Pareto front under the Shift/Swap neighborhood considered here.

V. Conclusions

This work has proposed the Pareto Plateau Connection Graph (PCG) as a metric to help assess problem difficulty from the perspective of a given search algorithm. It was shown that the analysis of small instances of three common combinatorial optimization problems provides insights in close agreement with much of what is known concerning the performance of certain classes of search algorithms on those problems.

Another possible line of research is to treat the data directly as a Markov chain and perform the analysis at that level rather than trying to visually analyze the graph structure. With complete information, this would allow us to derive expected bounds on the running time of certain types of algorithms. However, the estimation necessary to build the model of realistically sized problems raises serious questions for this sort of analysis. Nonetheless, it may be a fruitful direction of future research.

There remains a substantial amount of work to do. While

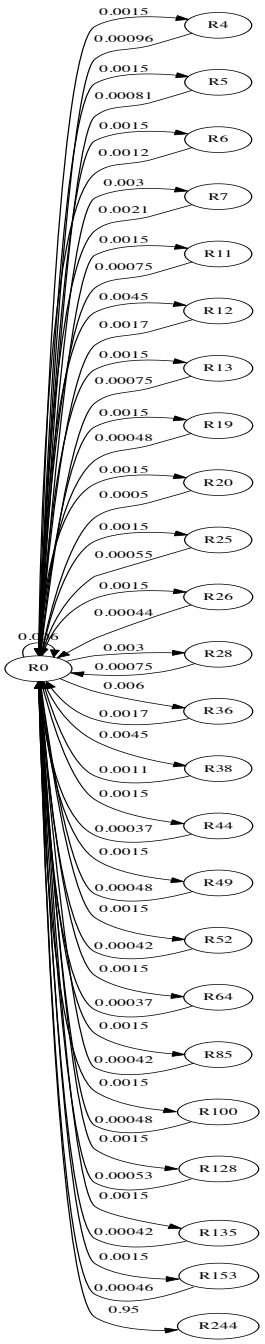

Fig. 7. Section of the PCG for a 3×17 biobjective mGAP instance at or near the Pareto front.

this work presented multiple mechanisms by which the type of large graphs which would result from realistic problems, it is still uncertain how much the information lost in these transformations will affect the ability to draw meaningful conclusions.

REFERENCES

[1] L. Alfandari, A. Plateau, and P. Tolla. A two-phase path relinking algorithm for the generalized assignment problem. In *Proceedings of the Fourth Metaheuristics International Conference*, pages 175–179, 2001.

[2] L. Alfandari, A. Plateau, and P. Tolla. A path relinking algorithm for the generalized assignment problem. In M. G. C. Resende and J. P. Sousa, editors, *Metaheuristics: Computer Decision-Making*, pages 1–17. Kluwer Academic Publishers, 2004.

[3] K. D. Boese. Cost versus distance in the traveling salesman problem. Technical Report TR-950018, University of California at Los Angeles, 1995.

[4] P. C. Chu and J. E. Beasley. A genetic algorithm for the generalized assignment problem. *Computers and Operations Research*, 24(1):17–23, 1997.

[5] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6:182–197, 2002.

[6] J. A. Díaz and E. Fernández. A tabu search heuristic for the generalized assignment problem. *European Journal of Operational Research*, 132(1):22–38, 2001.

[7] D. Garrett. *Multiobjective Fitness Landscape Analysis and the Design of Effective Memetic Algorithms*. PhD thesis, 2008.

[8] D. Garrett and D. Dasgupta. Analyzing the performance of hybrid evolutionary algorithms for the multiobjective quadratic assignment problem. In *Proceedings of the 2006 IEEE Congress on Evolutionary Computation*, 2006.

[9] D. Garrett and D. Dasgupta. Multiobjective landscape analysis and the generalized assignment problem. In *Proceedings of Learning in Intelligent Optimization II*, 2007.

[10] H. Hoos and T. Stützle. *Stochastic Local Search: Foundations and Applications*. Elsevier, 2005.

[11] J. Knowles and D. Corne. Towards landscape analyses to inform the design of a hybrid local search for the multiobjective quadratic assignment problem. In A. Abraham, J. R. del Solar, and M. Koppen, editors, *Soft Computing Systems: Design, Management and Applications*, pages 271–279. IOS Press, 2002.

[12] J. Knowles and D. Corne. Instance generators and test suites for the multiobjective quadratic assignment problem. In *Evolutionary Multi-Criterion Optimization (EMO 2003), Second International Conference*, pages 295–310, Faro, Portugal, April 2003.

[13] T. Koopmans and M. Beckmann. Assignment problems and the location of economics activities. *Econometrica*, 25:53–76, 1957.

[14] M. Laguna, J. P. Kelly, J. L. González-Velarde, and F. Glover. Tabu search for the multilevel generalized assignment problem. *European Journal of Operational Research*, 82(1), 1995.

[15] M. López-Ibáñez, L. Paquete, and T. Stützle. Hybrid population-based algorithms for the bi-objective quadratic assignment problem. Technical Report 1, FG Intellektik, FB Informatik, TU Darmstadt, 2006. Journal of Mathematical Modelling and Algorithms.

[16] H. R. Lourenço and D. Serra. Adaptive search heuristics for the generalized assignment problem. *Mathware and Soft Computing*, 9(3):209–234, 2002.

[17] M. Racer and M. M. Amini. A robust heuristic for the generalized assignment problem. *Annals of Operations Research*, 50(1):487–503, 1994.

[18] S. Sahni and T. Gonzalez. P-complete approximation problems. *Journal of the ACM*, 23(3):555–565, 1976.

[19] E. D. Taillard. Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17:443–455, 1991.

[20] M. Yagiura, T. Ibaraki, and F. Glover. An effective metaheuristic algorithm for the generalized assignment problem. In *2001 IEEE International Conference on Systems, Man, and Cybernetics*, pages 242–250, 2001.

[21] M. Yagiura, T. Ibaraki, and F. Glover. A path relinking approach for the generalized assignment problem. In *Proceedings of the International Symposium on Scheduling*, pages 105–108, 2002.

[22] M. Yagiura, T. Ibaraki, and F. Glover. An ejection chain approach for the generalized assignment problem. *INFORMS Journal on Computing*, 16(2):133–151, 2004.

[23] M. Yagiura, T. Yamaguchi, and T. Ibaraki. A variable depth search algorithm with branching search for the generalized assignment problem. *Optimization Methods and Software*, 10(3):419–441, 1998.

[24] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the strength pareto evolutionary algorithm. Technical Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, May 2001.